NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

A HOLISTIC MANAGEMENT ARCHITECTURE FOR LARGE-SCALE ADAPTIVE NETWORKS

by

Michael R. Clement

September 2007

Thesis Advisor:	Alex Bordetsky
Second Reader:	Karl Pfeiffer

Approved for public release; distribution is unlimited.

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.			
1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** September 2007	**3. REPORT TYPE AND DATES COVERED** Master's Thesis	
4. TITLE AND SUBTITLE A Holistic Management Architecture for Large-Scale Adaptive Networks		**5. FUNDING NUMBERS**	
6. AUTHOR(S) Michael R. Clement			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000		**8. PERFORMING ORGANIZATION REPORT NUMBER**	
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A		**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**	
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.		**12b. DISTRIBUTION CODE**	

13. ABSTRACT (maximum 200 words)

This thesis extends the traditional notion of network management as an indicator of resource availability and utilization into a systemic model of resource requirements, capabilities, and adaptable allocations from a services perspective. Central to this model is a mapping of user information requirements onto measurable network attributes that can be used to evaluate levels of service. A monitoring infrastructure suitable to capturing and visualizing these attributes is also investigated. The outcome is a framework for understanding, measuring, and monitoring informational services in terms of their effects on a network. These results could be used to develop semi-automated and adaptive network monitoring and management suites that would support large-scale network centric operations.

14. SUBJECT TERMS Adaptive Networking, Holistic Network Management.			**15. NUMBER OF PAGES** 103
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UU

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited.

A HOLISTIC MANAGEMENT ARCHITECTURE FOR LARGE-SCALE ADAPTIVE NETWORKS

Michael R. Clement
Civilian, Naval Postgraduate School
B.S., Seattle University, 2004

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN INFORMATION TECHNOLOGY MANAGEMENT

from the

NAVAL POSTGRADUATE SCHOOL
September 2007

Author: Michael R. Clement

Approved by: Dr. Alex Bordetsky
Thesis Advisor

Lt. Col. Karl D. Pfeiffer, USAF
Second Reader

Dr. Daniel C. Boger
Chairman, Department of Information Sciences

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

This thesis extends the traditional notion of network management as an indicator of resource availability and utilization into a systemic model of resource requirements, capabilities, and adaptable allocations from a services perspective. Central to this model is a mapping of user information requirements onto measurable network attributes that can be used to evaluate levels of service. A monitoring infrastructure suitable to capturing and visualizing these attributes is also investigated. The outcome is a framework for understanding, measuring, and monitoring informational services in terms of their effects on a network. These results could be used to develop semi-automated and adaptive network monitoring and management suites that would support large-scale network centric operations.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

First and foremost, I want to thank the Lord for His strength and guidance working in my life. All that I achieve in life is by His will. Ad Majorem Dei Gloriam.

To my parents, my family, and Caitlin: For supporting me, listening to me when I got frustrated, and encouraging me never to settle for "good enough," I thank you and I love you.

To all those who have taught me, inside and outside the classroom: Thank you for sharing your knowledge and wisdom.

To Brian Rideout, Andy Strickland, R.J. Simmons, Jamie Gateau, and all the students who have encouraged and supported me: Your insights, diversions during the stressful times, and hospitality during late-night study sessions kept me going throughout the program; thank you all.

To Dr. Netzer, and all the faculty and staff involved in CENETIX and TNT: Thank you for creating an outstanding work environment. The field experiments have been my inspiration and my motivation.

To Lt. Col. Pfeiffer: Thank you for allowing me to bounce my half-cooked ideas off you, for guiding me in performing quality research, and for a good cup of coffee in the mornings!

Finally, to Dr. Bordetsky: Sir, you have supported me in this endeavor on a level that has far surpassed my greatest expectations. Your leadership, guidance, and friendship made this achievement possible. I thank you from the bottom of my heart for taking me under your wing and building me up as a researcher.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. OVERVIEW

The modern military force is becoming increasingly networked, and what was yesterday an experimental supplement to existing operating procedures is today a fundamental capability. Unmanned Aerial Vehicles (UAVs) provide live Intelligence, Surveillance, and Reconnaissance (ISR) to localized forces on the ground; command centers located in the U.S. control ISR platforms thousands of miles away; and commanders monitor near-real time tracks of blue forces on the ground, in the air, and at sea. These capabilities are merely the tip of the iceberg. Developmental technologies for fusing geospatial, biometric, imagery, and other sensor data are providing levels of battlespace awareness previously unseen. All of these data sources need to be fed to various consumers for processing, analysis, and subsequent decision-making.

As the number of data sources and data consumers increase, even the most capable networks will be taxed to deliver all users' critical data in a timely and reliable manner. This prompts the need for mature network monitoring and management (herein collectively referred to as network management) technologies that enable administrators to monitor the usage of network resources and manage the allocation and prioritization of resources to different users' Critical Information Requirements (CIRs).

Current network management technologies are oriented toward the network administrator, who reasonably is most interested in metrics including bits or packets per second, link and path latency, and percent packet loss. However, these metrics only describe the aggregate effect of myriad data "flows" traversing each link that comprises the network. Industry standards such as the Simple Network Management Protocol (Case et al. 1990) are link- and path-oriented; the missing piece is the relationship between this layer of network performance and the application and services layers which describe the actual data flows that satisfy user CIRs.

In an ideal, futuristic network-centric operation, users would dictate their CIRs to the Global Information Grid (GIG) in real-time; these would be evaluated against available resources and information providers; and intelligent selection, prioritization, and adaptation would establish the best fit between all users' requirements. This mechanism would be capable of not only prioritizing certain users or applications over others, which is similar to modern Quality of Service (QoS) technology, but also to both adapt application traffic to fit within available resources and modify the behavior of the network itself.

This research work focuses on the "network awareness" aspect of this vision; namely, generating a holistic view of the network state, both in terms of capability and in terms of current load and user requirements. In particular, this thesis aims to elucidate the relationship between an operator's view of a "service" and the network administrator's view of "network performance," and build a framework for describing network performance in terms of the specific services fulfilling user requirements, enabling both descriptive and predictive analysis at the services layer.

B. MOTIVATION FROM NETWORK-CENTRISM AND SYSTEMS THINKING

A major premise of this research is that effective network management is a critical element of Network-Centric Warfare (NCW). (Keshav and Sharma 2000) postulate that service quality is closely tied to the quality of network management. This view is reflected in core NCW literature, including (Alberts et al. 1999, 191):

> An infostructure must be properly managed to ensure that it is dynamically tuned to meet the warfighter's needs. Enhanced capabilities for network operations will provide operational commanders with a real-time picture of the status of the backplane. This picture, when combined with advanced capabilities for intelligent network management, will provide commanders with the flexibility to tune the infostructure and synchronize information transport and processing with military operations.

Alberts et al. describe a "real-time picture" that aligns with the holistic network view discussed above. Their model exposes some of the requirements of such an instrument. The need for management implies that a network is an inherently scarce

resource; thus the role of a network-centric management system is to mediate all users' CIRs against the resources available for fulfilling those CIRs. To do so, the capability, requirements, and usage of the network must be monitored in as near real-time as possible. Changes may occur so quickly that a comprehensive but high-latency monitoring tool will never yield timely and actionable information. This drives a requirement to find a suitably minimal set of data inputs to feed the network management model, minimizing the transmission and processing overhead required for management.

The challenges of building models to describe dynamic systems are well-known to the field of systems theory. Unlike detailed complexity, where with sufficient data input and processing time nearly any calculation is possible, system thinking treats the existence of complex relationships between variables that make analytics difficult (Senge 2006, 71):

> But there is a second type of complexity… situations where cause and effect are subtle, and where the effects over time of interventions are not obvious. Conventional forecasting, planning, and analysis methods are not equipped to deal with dynamic complexity.

It may not be feasible to build a comprehensive, analytical model for the behavior of data networks. Not only does each node, link, and service have myriad descriptive parameters; the quantity, combination, and configuration of these components extend indefinitely within the bounds of physical and logical limits. The overabundance of potential data input greatly increases the challenge of finding a simple approach to assessing the state of the network. Moreover, the performance state of one network link may be dependent on traffic originating and terminating in distant parts of the network, adding to the subtleties of interaction. Finally, as pointed out in (Barford et al. 2001), each location in the network sees only a "projection" of the whole, making comprehensive data collection still more challenging. These obstacles indicate the need for a less comprehensive-analytical, more systemic-holistic approach to managing networks.

This approach might be summarized in the term *holistic network configuration management*: a model of network services and resources that culminates in the depiction

of network behavior as it relates to its intended use. The Fault, Configuration, Accounting, Performance, and Security (FCAPS) management model (International Engineering Consortium 2007) approaches this concept by defining the overarching business process of network operations, focusing on the reduction of system and network downtime and maximizing availability of information resources. In this research, the idea of maximizing user levels of service is re-investigated, starting from current implementations of network management and drawing on emerging research as discussed in the next chapter to produce a new model of configuration management from a holistic perspective.

C. ISSUES IN TODAY'S SOLUTIONS

Modern network monitoring platforms focus primarily on two facets of network performance: reach and link performance. Reach refers to the ability of a given networked node to converse with other nodes within the same network. Ideally, any networked node can reach any other node; however, reach may be restricted for many reasons, including overloaded links, weak wireless link signals, improper device configuration, or device failure. Link performance, as stated earlier, is the aggregate effect of all traffic flows traversing a physical network link relative to the capabilities of that link. This can be further divided into range and responsiveness, reflecting the three dimensions of telecommunications services as defined in (Keen and Cummins 1994).

When the network administrator is responsible for monitoring backbone infrastructure and the main concerns are accessibility of resources and gross resource utilization, these are appropriate aspects to monitor. However, when the network administrator is tasked with managing a rapidly-changing network both in terms of users and applications as well as the physical and logical topology, it becomes crucial to achieve a high-level understanding of network behavior and activity. Although existing network management solutions may be able to answer the who, what, and where of resource usage, none tie these back into the larger picture of describing and monitoring the needs of network users: their CIRs.

The same shortfalls apply to modern QoS techniques, most of which are accounting-focused. These approaches operate by dividing a static-sized available resource among a known set of nodes and services, ensuring that at no time are more resources obligated than are available. While this does accomplish the goal of managing finite resources, even prioritizing certain nodes and services over others, it entirely misses the point of a service-focused network. The centerpiece of the network should be the users and their CIRs, rather than the links that service those users. Contrary to the QoS model, it may not be reasonable to guarantee that resources obligated at one time will remain available at subsequent times; applications may need to adapt to changes in the network environment. Likewise, if certain CIRs are overtaxing the network, it may be the responsibility of the network to adapt to better accommodate those needs. In order to inform such an adaptive model, the bar must be raised on network management strategies to provide a stronger user- or service-focus.

D. THE PATH AHEAD

Motivated by the concepts of network-centricity and system dynamics, and understanding the gaps in modern network management methodologies, the goal of this thesis is to develop a conceptual and prototypical framework for holistic network management. More specifically, the aim is to create an integrated picture of services and infrastructure, where resources, services, and user needs intersect to create a usable, actionable depiction of network behavior and a means to assess its performance in terms of articulated CIRs. This work will provide a foundation for further research in adaptive networking and holistic network configuration management.

One major gap that must be addressed is the relationship between a CIR and its underlying applications, and in turn between those applications and their effects on a physical and logical network. This research builds on existing studies of applications performance and layered network models, and leads to a framework for describing CIRs at a high level and translating those descriptions into measurable network properties. Using this framework, it will be possible to develop a measurement methodology that provides satisfactory awareness of CIR performance with minimal overhead. The product

of this thesis is a conceptual model of CIR description and translation, and discussion of measurement and visualization techniques appropriate to a holistic network management approach. This model should allow a network administrator to achieve awareness not only of how the network is behaving, but of how it is being used and how well it is satisfying its use.

II. LITERATURE REVIEW

A. OVERVIEW

The overall goal of this research is to translate operators' CIRs (i.e., information requirements expressed from the viewpoint of the user) into concrete network requirements to be assessed against the live tactical network. The results must be gathered and presented in such a way that will neither overload the network nor overwhelm the user. To achieve this, both a model for translating information requirements and an appropriate measurement architecture must exist; although the measurement architecture is not defined by this research, its subsequent discussion necessitates a review of prior work. This chapter will review related academic literature as well as the contributions of industry, identifying both the foundations and the gaps that form the starting point for the contributions of this research.

B. REPRESENTING USER'S REQUIREMENTS

A major challenge of effective service-level network management has always been how to describe services as the end-user perceives them. This problem is identified repeatedly in academic literature (Parulkar et al. 1997) (Galetzka 2004). In attempting to build a management architecture, Parulkar et al. (1997) point out the difficulty of dynamic network adjustment due to "the fact that demands keep changing and are not completely known." Fortunately, there is significant existing work in translating requirements at one layer into measurable properties at another.

Research into Quality of Service architectures deals heavily with the mapping of services across layers. The International Telecommunications Union (ITU) has published several "recommendations" documents on this topic. ITU-T Recommendation G.1010 (International Telecommunications Union, 2001) is referenced heavily in academic literature; this document describes several common types of traffic flows, ranging from video and audio streams to email and web browsing, and for each provides expected throughput as well as tolerance to packet loss, errors, latency, and jitter. This and similar

studies form a foundation for mapping application-specific traffic to generic network attributes, which is useful when the set of services is known in advance.

Other research focuses on models for translation from application to network requirements. Nahrstedt and Smith (1994) present a model of network management based on requirements translation starting at the application layer; however, they feed their result into a QoS control loop rather than a monitoring capability. Their model defines in detail the relationships between various multimedia attributes (e.g., video resolution and frame rate) and complementing network attributes (e.g., bit rate, latency, and jitter). While their work is not exhaustive, it provides an excellent starting point for further quantitative study of these relationships. DaSilva (2000) complements their model with a study of QoS for packets where delay and loss are introduced at the data-link layer. He argues the importance of understanding how network behavior at lower layers affects the ability to guarantee service levels at higher layers.

There has also been work done mapping the application layer onto higher layers, closing the gap on describing services from the human perspective. Guo and Pattinson (1997) define a four-layer model consisting of network, system, application, and user, and identify five categories of quality requirements spanning from traditional QoS metrics to subjective human-based qualities. Bauer and Patrick (2004) reference this model and build their own that extends the seven-layer OSI model (Zimmerman 1980) with three additional layers: human interface, human performance, and human needs. Although neither paper specifies the mechanics of these relationships, they create a basis for phrasing application and network requirements in terms of the user's needs. As Bauer and Patrick point out, these extensions go beyond the definition of QoS, into what many researchers have termed Quality of Experience.

Quality of Experience (QoE) is a term found in recent literature that discusses service-oriented measures of quality and performance. Although the precise definitions vary, QoE is in general terms the perceived experience by the user of the services that user expects to receive. A white paper by Polycom (O'Neil 2002) stresses that QoE "is the true litmus test" of an end-user's experience. Since the typical end-user does not articulate service requirements in terms of bit rate or jitter, much of QoE research focuses

on enabling users to dictate their own requirements to the network and to interactively prioritize their needs. Siller and Woods (2003) discuss a resource arbitration system based on QoE, and conduct experiments using a tunable-knob approach wherein users manually vary parameters at the application and network layers to achieve the QoE they desire. Galetzka, on the other hand, builds a model for what he terms "user-perceived quality of service" (Galetzka 2004), which he relates to QoE. His model ties together the inter-layer mappings and effects as discussed earlier in (Nahrstedt and Smith 1994) and (DaSilva 2000) and the categorizations from (International Telecommunications Union 2001), and creates four user-layer attributes that apply across all service types: availability, timeliness, accuracy, and affordability. He presents an example for a television programming guide service, defining these parameters within that specific context; however, he does not propose a generic framework for service parameter translation.

Although models and examples abound, in order to implement QoE or other service-layer metrics on a large-scale network such as the Internet or Global Information Grid, there must be a general framework for describing services in terms of their requirements on underlying layers. Zhou et al. (Zhou et al. 2005) propose DARPA Agent Markup Language for QoS (DAML-QoS), an ontology for service semantics. DAML-QoS is oriented toward matching web services with customers' needs; each service request includes attributes such as service cost or response time, which are constraints that responding service providers must be able to satisfy. Their framework includes a notional measurement architecture for evaluating web services according to these metrics, and an ontology converter and reasoning engine to match available services to service requests. XQoS (Exposito et al. 2002) is another format for service specification; in their case, an XML schema that describes QoS requirements. For each component of a multimedia stream, the schema defines characteristics such as minimum reliability that must be met by the network. However, both approaches still rely on a heavy knowledge of network and application characteristics, leaving the translation from a true user-centric layer to the underlying layers an open field for study.

Regardless of the mechanics, once CIRs are translated into requirements for the network, these requirements must be monitored effectively so that the human operator or administrator, or an autonomous agent, can appropriately adjust the behavior of applications and of the network to suit the needs of all users. In order to accomplish this, the network must be outfitted with a suitable measurement architecture.

C. THE GLUT OF MEASUREMENT

When it comes to the broad field of monitoring network performance, there is no lack of metrics to assess and tools with which to capture those metrics. From polling of individual devices through SNMP (Case et al. 1990) to application traffic analysis with NetFlow (Claise et al. 2004), to active probing of network topology and link capacity (Jacobson), the network administrator might easily be overwhelmed with all the network performance data available. However, this assumes the right infrastructure is in place to capture that data. There are several approaches to network measurement and monitoring, each with its own benefits as well as costs.

A classic approach to network measurement is device polling or probing: periodically asking each element of the network if it is alive, and for information about its state. This began with the Ping utility (Kessler and Shepard 1997) which assesses two metrics: whether or not the node being polled is alive, and how long it takes to send a message to that node and receive a response. Interestingly, these are still very common metrics used by network management suites including Solarwinds (Solarwinds). Variants of this measurement include Traceroute (Kessler and Shepard 1997) and Pathchar (Jacobson 1997), which measure the path taken between two nodes and the per-hop link capacity along that path, respectively.

One step beyond basic probing techniques are device statistics-polling protocols. The Simple Network Management Protocol (SNMP) (Case et al. 1990) is a current standard for per-device data collection. SNMP provides access to each device's Management Information Base (MIB), which is a collection of configuration fields and counters kept by the device. The interface MIB (McCloghrie and Rose 1991), which is maintained for each network interface on a device, has values such as packets received

and sent, bytes received and sent, and packet error counts. Proposed extensions including the SM MIB (Choi and Hwang 2005) monitor additional end-to-end path characteristics including packet loss and jitter. Solarwinds and other network management tools monitor SNMP data, providing one means of tracking the performance and behavior of the network.

However, there is more to the network than what node and link measurements depict. These statistics do not reflect the behavior of individual applications. Cisco's NetFlow standard (Claise et al. 2004) and Foundry's sFlow both capture statistics on each flow traversing a router or other capture device. These statistics may then be analyzed and presented via collection systems such as Ntop (Deri and Suin 2000). Not only does this provide another dimension of network performance data, it is also a departure from the centralized polling model of SNMP and its kin. In the NetFlow architecture, each monitoring device captures and aggregates network data, periodically sending reports to a collection server. This has the benefit that if no traffic is traversing a part of the network, no reports are generated.

Several researchers postulate network measurement models that incorporate these and other techniques to create a coherent view of network activity and behavior. Keshav and Sharma (Keshav and Sharma 1998) propose a lifecycle model of the network that includes topological discovery, monitoring, performance problem identification, configuration testing through simulation, and modification leading to a new network. Aside from simulation and modification, this model draws on all the techniques identified above; in fact, the authors propose integrating several existing tools to achieve the necessary data collection. Researchers at AT&T (Caceres et al. 2000) put forward another integral approach, in this case collecting ICMP, SNMP, NetFlow, and server log data into a data warehouse for wide-scale network analysis. Most of their research applications, however, are oriented toward data mining vice real-time monitoring of services.

Other models have been proposed that diverge from the use of off-the-shelf tools, instead favoring a customized approach. Parulkar et al. (1997) lay out an architecture that, on the surface, looks similar to (Caceres et al. 2000) in that distributed data captures are gathered in a central data store. However, this particular approach is geared toward

real-time visualization and monitoring of network behavior. As such, they design dedicated probing nodes for data collection, distributing the need for processing power and reducing network overhead by only transmitting processed reports across the network. Estan and Varghese (Estan and Varghese 2003) recognize the scalability issues of tracking millions of discrete flows via technologies like NetFlow, and propose a scheme that only monitors flows over some threshold of link utilization. Their assumption is if there is a congestion point in the network, then it is most likely occurring because of large flows; therefore, tracking the behavior of these flows will lead to the problem area as effectively as tracking all flows, and with significantly reduced overhead.

One common theme in nearly all approaches discussed above is resource overhead in network monitoring. For any measurement instituted, there is inherent processing, memory, and network utilization attached. Barford et al. (Barford et al. 2001) investigate this overhead and come to the conclusion that, given a relatively stable "core" infrastructure, increasing the number of network probes does not yield more useful network information. They focus on Traceroute, which is a relatively lightweight measurement, and demonstrate that for very large networks there is little gain in surpassing a few well-placed probes. Similar to the resource-conscious architecture in (Estan and Varghese 2003), this paper establishes an additional constraint on network monitoring architectures: they must not impinge on resources needed to conduct mission-critical operations.

D. CONCLUSIONS FROM THE REVIEW

There is an abundance of interest both in translating user requirements into resource requirements and in measuring the network to evaluate the satisfaction of those requirements. Despite the depth of research already done in these areas, there is still significant ground to cover. There are many models for "Quality of Experience" and service-layer translation, but this work has mostly resulted in simple experiments or one-off examples; a generic framework that applies across service domains is still needed. It is also clear that while many network measurement tools exist, one must choose carefully the set of metrics to capture in assessing service satisfaction; not all measurements add

value, but all add overhead. In the following chapters, these topics will be further addressed, and a framework will be laid out for generic service description, translation, and measurement within the tactical network domain.

THIS PAGE INTENTIONALLY LEFT BLANK

III. DATA AND METHODS

A. CONCEPTUAL UNDERPINNINGS

From studying the strengths and weaknesses of existing network management models, some general requirements for a holistic model emerge, all centered on the idea of evaluating levels of service achieved vice network performance. To build a meaningful framework for service translation, measurement, and evaluation, it is important to formalize these requirements and define some concepts that will form the building blocks of this model. From these building blocks will emerge the framework for translating CIRs into attributes that can be measured and evaluated.

1. Requirements for a Holistic Network Management Model

There are five requirements this research identifies as crucial to creating the systemic-holistic (herein referred to as holistic) network management model described in the first chapter. The first three are functional requirements, which specify the objectives the model must be able to accomplish. The last two are quality requirements, which describe the way in which the model must work in order to be usable. Together, these requirements form the rationale for the architectural choices defined later in this chapter.

First, the model must focus on the information needs of the individuals using the network, i.e., the set of CIRs. As opposed to models that focus on the accounting of each traffic flow or the utilization of each link, a holistic model must evaluate the capability of the network to deliver the services requested by its users, and present network performance problems in terms of the affected CIRs and all relevant portions of the network. Disparate CIRs that share underlying infrastructure may cause compound problems; this model should identify all services and network components related to a problem so that administrators can effectively pinpoint and correct the problem.

Second, the model must be *descriptive*; that is, it must accurately depict the behavior of a network, given some set of input parameters. These inputs should be

sufficient to express the configuration of the network, i.e., its topology and capacity, and the activity on the network, i.e., the set of traffic flows traversing the network. Given these inputs, the model must correctly assess the performance of each CIR. In this case, configuration and activity data come from the measurement infrastructure, which is discussed later.

Third, the model must be *predictive*. It must accurately assess the effects of proposed changes to the current, known network state. Here, all statements pertaining to the model's descriptive capability apply; however, proposed changes must also be expressed as input parameters. This applies both to changes in network configuration, e.g., altering the characteristics of a link, and to changes in network activity, e.g., adding another CIR. The model must be able to provide the same resulting analysis for this hypothetical case as it would for the observed network.

Fourth, the model must be solvable within a useful decision-making time cycle. In order to be usable, the model must provide updated information to the administrator within the human response cycle necessary for maintaining a serviceable network state. Put another way, it must be able to inform the individual about emerging conditions quickly enough that the individual can react and maintain acceptable levels of service. Van Creveld (1985) discusses this relationship between speed and effective command of forces; this carries over into the command of systems supporting those forces. This requirements places constraints on the types of models that may be used: certain types of simulation require many iterations to converge or predict a result; these may be inappropriate for generating high-speed results. There are also tradeoffs to be considered in the precision of the model versus its responsiveness.

Finally, use of the model must not impinge on operational traffic. There is a wealth of measurement and assessment techniques available which may be leveraged to obtain any number of network performance metrics. However, as alluded to in (Barford et al. 2001) capturing and processing every possibly metric is both pointless as many metrics do not add value to the assessment, and obstructive as each additional metric collected detracts from the resources available to services. The model should only utilize measurements and computations that are necessary for accurately describing and

predicting service performance, and may make use of techniques such as "no news is good news," only sending pertinent changes in network state.

2. Defining the Building Blocks

Analytical techniques break systems into their component parts, investigating the properties of each part separately. Systems thinking starts with the parts and studies the relationships between them that form the whole. Following in the latter tradition, this model begins with the definition of its elemental properties, and builds from there.

If the "Service" or CIR is the highest concept in this model, the lowest is the notion of "Bits in Time." This model proposes that all other network properties can be expressed in terms of X bits in Y time. A network requirement would then take the form of needing to communicate X bits within Y time. Bandwidth is a measure of the X bits per Y time, on average. Latency is the Y time it takes to communicate X=1 bit. Loss is $X_T - X_R$ bits over Y time, where X_T is the bits transmitted and X_R is the bits received. The Bits in Time relationships for common network attributes are presented alongside with definitions based on relevant literature in Table 1.

Attribute	**Common Definition**	**Bits in Time Definition**
Throughput	Average rate of bits between source and destination	Average X bits in Y time
Burstiness	Variation in rate of bits between source and destination	Distribution of Y time between X bits
Latency	Average time taken for bits to arrive at their destination	Average Y time per bit
Jitter	Variation in time taken for bits to arrive at their destination	Instantaneous Deviation in Y time per bit
Loss	Percent of sent bits that do not arrive at their destination	X_T bits sent minus X_R bits received over Y time

Table 1. Network Attributes Defined as "Bits in Time"

This is similar to but distinct from the Service Level Agreements (SLAs) defined in (Clark and Gilmore 2006), which are of the form X percent of bits (or messages) in Y time. One of the fundamental notions of the Bits in Time model is that, assuming some level of connectivity and given enough time, all bits will get through. If reliable protocols are implemented, then any "lost" bit can be rescheduled and retransmitted, merely delaying the reception of that bit. Therefore loss is only characterized for bounded time. This introduces the second concept: each bit has value, and that value is time-dependent. In fact, it could be said that each bit's value is a monotonically-decreasing function of time, which approaches zero at some critical "value deadline," after which the bit no longer has value and may as well not be transmitted at all. Loquinov and Radha (2001) discuss this in detail as it applies to streaming video.

Some simple examples can illustrate. A meteorologist may wish to forecast tomorrow's weather; however, if the best model can compute that forecast in no fewer than forty-eight hours, that forecast will be of no value since the tomorrow's weather will be known empirically by that time. Some streaming video protocols transmit each frame in a separate packet; if a packet misses its deadline for decoding then that frame is dropped and the video continues to play. Even if the packet was merely late to arrive due to high latency, it was effectively lost because it did not arrive while it still had value. Hence, loss is characterized within a finite timeframe, because bits have time-dependent value.

The notion of bit-value has another meaning. If each CIR has value to an individual and each CIR is composed of bits carrying information, then each bit must also have some value to that individual. This relationship is complicated by the specific type of information the bit represents. As will be discussed later, different types of information have varying tolerances to loss, latency, et cetera. A missed bit in a video stream is of lesser consequence than a missed bit in a text message, indicating that the value of single bits is both information type- and context-dependent. However, collections of bits may still be quantified, or at least qualified, in terms of the value they potentially deliver to the recipient.

3. Links and Flows: The Basic Units of Network Configuration

The physical-logical network consists of nodes, e.g., computers and routers, connected by links. Since modern computing systems process data at speeds far greater than typical network links, performance attributes need only be measured for links. Data travels from source to destination along a sequence of links, known as a path. The path has certain performance attributes that are based on the links that comprise it. For instance, the maximum one-way throughput of the path is theoretically the minimum of the throughputs of all links along that path. Path latency is theoretically the accumulation of all link latencies, plus minute overhead introduced by intermediate nodes. These aggregate performance attributes in turn affect the other basic unit of networks: flows.

Flows are the applications network equivalent of links: they are the singular components which represent each independent stream of communication. Aggregates of one or more flows form a *service*, which is the instantiation of a CIR. Each service has value to the recipient of that service; likewise, each flow has a value to the service. Flows have performance attributes, here in the form of requirements, which utilize the resources of the paths along which those flows travel. Hence, there is a hierarchy of related attributes and requirements spanning from services to links, as depicted in Figure 1.

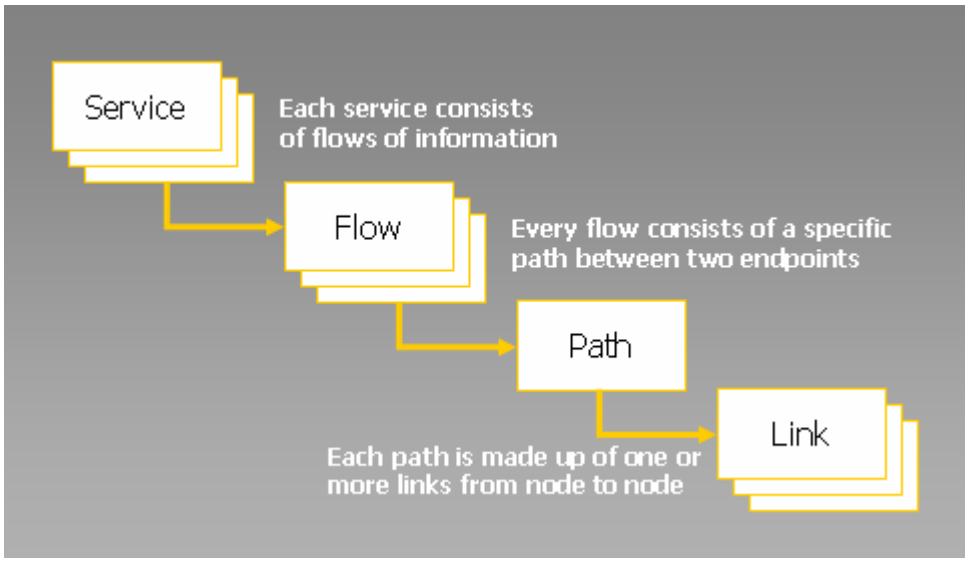

Figure 1. Hierarchy of Application and Physical Network Components

Each CIR is instantiated as a service, which in turn consists of distinct flows that carry information between endpoints on the network. These flows each traverse a specific path, which is comprised of links. Here the crucial point of comparison emerges: the network aggregate, paths, are attributed with certain capabilities; the services singular, flows, are attributed with certain requirements. Services have value to the individual, meaning that the capability of each path to effectively carry its respective flow affects the manifest value of the service. Taken one step further, each link, which may carry many flows along many paths, and which has finite capacity, affects the performance and therefore the value of each service it bears. The ability to evaluate the performance of services in light of these relationships will yield a holistic picture of both network and service performance.

Such an approach has certain difficulties associated with it. The relationship between each flow-bearing path and the set of all underlying links is a complex mapping. Distinct flows between pairs of endpoints in seemingly distant parts of the network may share common links, putting those flows in competition for finite resources. Tabulating each combination of service, flow, link within that flow's path, and every network attribute would take a matrix of several dimensions. Memory and computational requirements for evaluating both descriptive and predictive cases would quickly grow out of reasonable proportion. The fundamental notions of Bits in Time and bit-value deadlines, along with a language for describing CIRs in terms of component flows and those flows' requirements, offer the possibility of a simpler way of expressing services and evaluating service performance. Building this language is the next requisite step in framing this model.

B. TRANSLATING CIRS INTO BIT-TIME REQUIREMENTS

Each CIR is an expression of an individual's specific informational need over a particular period of time. It might be a video teleconference between generals, or the position and status of a neighboring fire team. If written in plain English, CIRs would be simple statements such as "live surveillance video from a Predator UAV flying over Al-Anbar province from 1900-2300." Explicit in these statements are data sources and

destinations, data types, and the time at which the CIR is required. Yet other important parameters remain implicit, such as value of this CIR to the requestor, component flows that constitute the service, and the relationships between network performance of the flows and the overall performance of the service. To make these parameters explicit, each CIR must be expressed in a definite language that consists of key terms understood both by operators and the network management system.

The following scenario is used throughout this section to demonstrate the concepts of the proposed model: a watch-stander in the field wishes to monitor video surveillance from a UAV flying over a border area; the watch-stander needs to see any humans or vehicles attempting to cross the border. The UAV runs on auto-pilot while airborne, leaving the watch-stander responsible for controlling a gimbaled camera to sweep the border as the UAV flies overhead.

1. Breaking CIRs into Flows and Service-Level Attributes

A computable expression of a CIR must articulate both service-level requirements and descriptions of each component flow. Each flow must be described in terms of its performance requirements and the relationship between its performance and the value achieved. For instance, if the CIR for the scenario above is assessed and the video is providing two frames per second at low resolution, its value to the recipient may be lower than if video was providing 20 frames per second at high resolution.

Requirement	Attribute
How quickly recipient gets a file, email, etc.	Responsiveness
How near a recipient's multimedia stream is to real-time	Responsiveness
How easily heard or seen is the party on the other end of a teleconference	Clarity
Ability to see movement and motion in video	Clarity
Ability to make out fine details in video or audio	Clarity
Not missing any important messages	Reliability
How frequently a unit's status is updated	Responsiveness
How quickly a question is answered	Responsiveness
Not encountering hang-ups in a multimedia stream	Reliability

Table 2. Typical Service-Level Requirements Expressed in CIRs

Table 2 lists examples of high-level service requirements along with possible service-level attributes. These are the kinds of attributes the individual specifying the CIR may use to articulate requirements at a high level. Notice that these service-level attributes differ from network attributes such as throughput and loss, though they may map onto one or more network attributes. An important difference is that service-level attributes express qualities that the user experiences, whereas network attributes express underlying qualities that must exist to provide that experience. The way in which service-level attributes map to measurable network metrics may be context-dependent and should be transparent to the user.

Immediately, some of these attributes apply to the scenario. First, the watch-stander likely cares that the video is near real-time, because watching video of a vehicle crossing the border several minutes after it actually happens is significantly less useful for tracking and intercepting that vehicle. Responsiveness is also important in a second way: the camera control messages sent by the watch-stander to the UAV must get through quickly in order to have a usable, interactive surveillance platform. The video must have clarity; being able to see the motion and details of people and vehicles on the ground is more useful than seeing small, unrecognizable blotches. Finally, there is a

requirement for reliability insofar as the video must be relatively smooth and have a low frame-drop rate. However, this requirement is not as strict, as some loss is acceptable for streaming video. Service-level attributes for the scenario CIR are shown in Figure 2, along with notional mappings onto network attributes.

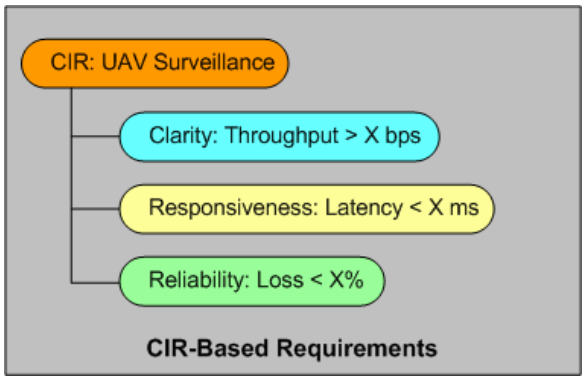

Figure 2. Service-Level Attributes for a CIR

The scenario CIR may consist of a few distinct flows: a one-time transaction to authenticate with and negotiate for control of the UAV camera; a continual stream of video; and periodic messages to control the direction and zoom of the camera. Each flow has performance attributes which aggregate to determine overall service performance. Since authentication and control negotiation will likely only occur once at the beginning of the service period, its responsiveness requirement is not very strict. However, it does have a strict reliability requirement; those bits must get through before the next step can proceed. On the other hand, the video stream itself can suffer some loss, but should be more responsive to guarantee near real-time coverage. Camera control has perhaps the most stringent requirements: it must be responsive as well as reliable. Slow response times will degrade the interactive quality of the system, whereas missed or erroneous messages will cause errant system behavior. Figure 3 shows service-level requirements mapped onto individual flows.

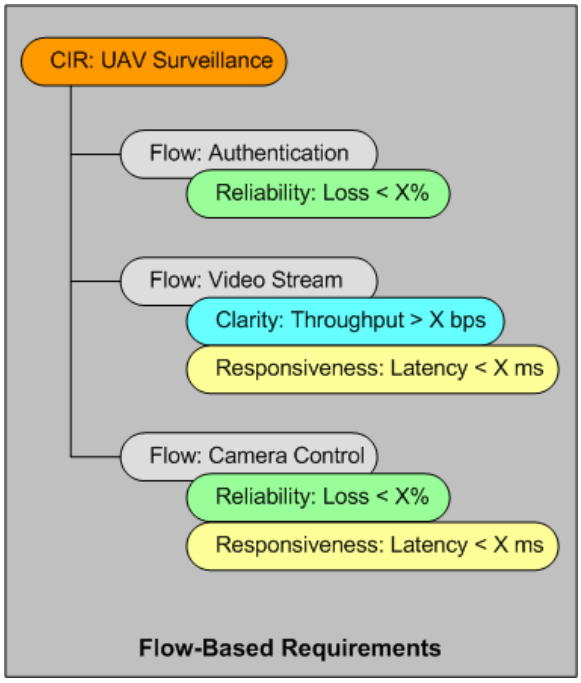

Figure 3. Breakout of Service-Level Attributes for Flows within a CIR

2. Mapping Flow Attributes onto Network Attributes

It is the task of the translation framework to map these service-level attributes onto measurable, network-level metrics. Some of these relationships are complex; for instance, clarity at the service level is a function of resolution, frame- or sample-rate, and distortion or bit-error. These in turn map onto network attributes such as throughput and loss. However, other factors including delay and jitter contribute to increased loss and decreased throughput, complicating these relationships. Hence, understanding these relationships is critical to translating CIRs.

There are several dimensions along which to distinguish types of flows. One is the burstiness of the flow, i.e., if it is singular, transactional, conversational, or streaming. A single DNS query would be transactional, whereas a voice call would be conversational. Most video flows are streaming, and singular flows apply only to one-off messages that require no response. Flow burstiness correlates with other attributes such as latency and loss. Singular and transactional flows tend to require low or zero loss, but often have more lenient latency requirements. Conversational and streaming flows such as voice

calls and video streams, tend to have higher tolerance to loss, but may need to have lower latency in order to maintain a near real-time requirement. (International Telecommunications Union, 2001) provides an elegant summarization of these dimensions in their own terminology in Figure 4.

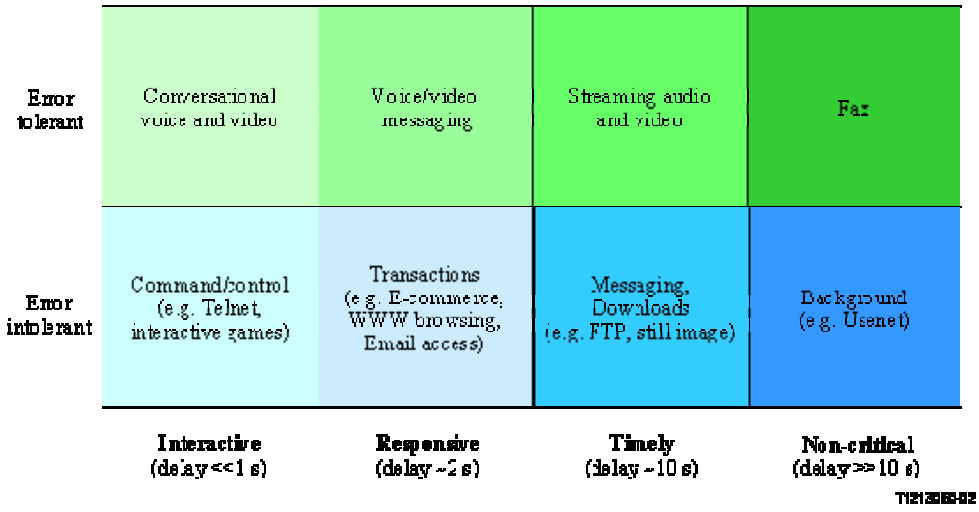

Figure 4. Model for User-Centric QoS Categories (From: International Telecommunications Union 2001)

The importance of other network attributes is less obvious, but they can be significant factors for some flow types. Jitter, the variation in latency, can cause individual packets to be exceptionally late to arrive and therefore miss their deadlines. Jitter has little effect on singular and transactional flows, but for streams that have real-time requirements jitter leads directly to loss. Loguinov and Radha (2001) demonstrate that for low-bitrate streaming flows, jitter has nearly two orders of magnitude more effect on perceived loss than actual packet loss and constant latency combined.

The performance attributes of each flow place requirements on the underlying network paths and their constituent links. In the scenario, all flows likely traverse the same path, but this is often not the case. Regardless, for each flow-bearing path, flow-level requirements must be evaluated against available resources in order to determine service performance. In descriptive cases where all services are directly observed, this may be a simple matter of aggregating available performance data. One approach would be to accumulate the "sum" of all flow requirements, then compare this against the

aggregate of the capabilities of all links along the flow path. This is demonstrated for three typical network attributes in Figure 5.

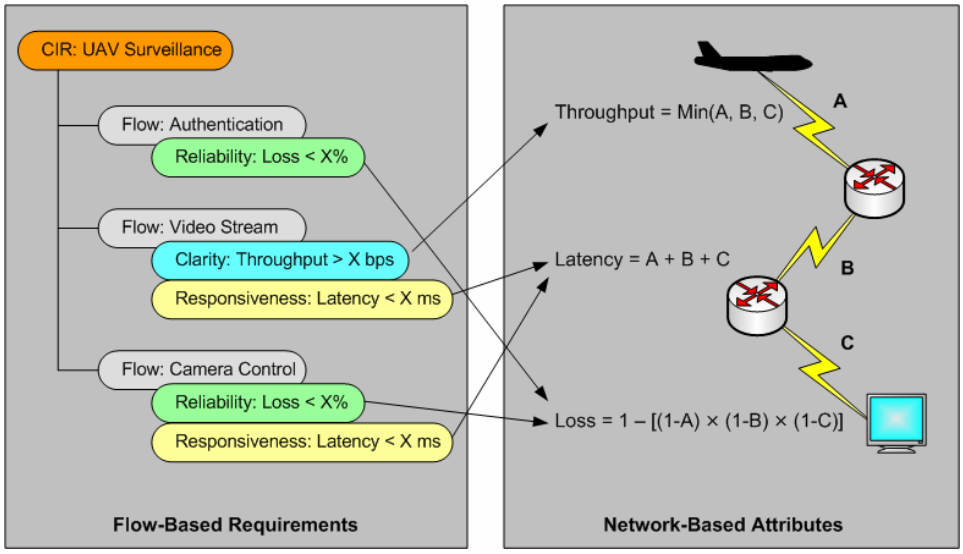

Figure 5. Mapping onto Typical Network Attributes

First, individual flow requirements must be aggregated and mapped onto the links and paths carrying those flows. Total throughput might be the sum of each flow's required throughput; total latency and loss might be the minimums of all flows' respective requirements. These requirements are then compared against the aggregate capability of the path. Figure 5 shows possible calculations of path attributes.

This approach does provide a usable hierarchy from service-level attributes down to network performance metrics. However, it does not achieve the elegance desired for a holistic model, nor does it account for the achieved value of the service. A model that drills down to the essential dimensions common to all types of flows will yield a translation mechanism more suitable for a holistic management model.

3. From Network Attributes to Bit-Time Curves

A different formulation of network requirements uses distributions of bits over value deadlines. Rather than using typical network attributes which describe the average behavior for traffic flows, these distributions or "bit-time curves" describe the statistical behavior of the entire flow. Along the time axis (where zero is the time of bit

transmission) is a curve that represents the number of bits that must successfully transit the network by each time *t*. This forms a snapshot from the perspective of the source for any given moment, depicting for all bits originating at that moment the distribution of value deadlines. For flows that include both bits that must be received very quickly and bits that may take longer to arrive, the bit-time curve will more accurately represent the flow's requirements than stating a single average throughput or latency requirement. An example bit-time curve is shown in Figure 6.

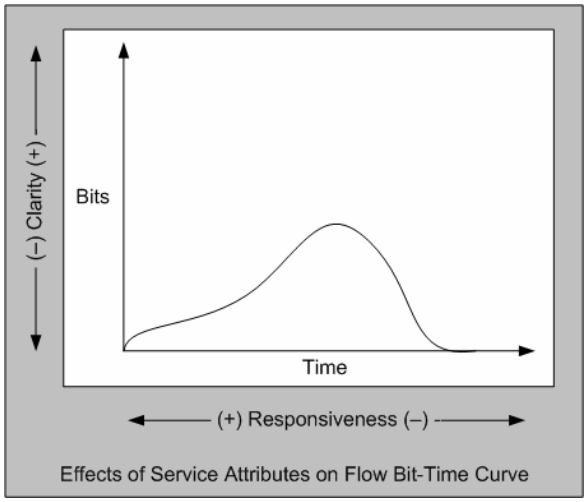

Figure 6. Example Bit-Time Curve for a Single Flow

This representation follows suit with the Bits in Time model; the focus of network measurement becomes the time it takes for bits to reach their destination. A latency requirement is a requirement that bits do not arrive past a certain deadline. Likewise, a loss requirement is a statement that no more than a certain percentage of bits may fail to arrive before their deadline. All flow traffic is then viewed in terms of value deadlines: a latent bit becomes a lost bit if it misses its value deadline; a lost bit, if retransmitted successfully before its deadline, is merely late.

Service-level attributes have effects on these curves. Figure 6 illustrates the relationships between some attributes and the shape of the curve. In order to increase clarity, more bits must be used to represent the information more precisely. This stretches the curve on the bits axis. Strict responsiveness requirements tighten the curve on the time axis. Reliability is more difficult to represent in the flow's curve, but will be

represented in link capability. These curves may be derived mathematically, based on the relationships between network metrics and service-level attributes already understood qualitatively. However, bit-value curves can also be derived from empirical data and generalized by flow type and service-level requirements.

The bit-time representation also allows for an elegant algebra for aggregating flow requirements into service requirements and link capabilities into path capabilities. In the case of flows, the aggregate of several flow requirements is the addition of their bit-time curves, as shown in Figure 7. Since each flow's curve describes the quantity of bits that must arrive by certain deadlines, adding flows together simply increases the quantity of bits due at their respective deadlines. This method can be used to determine the total capacity required of a link or path to support a given set of flows. It is important to note that, in practice, adding flows' curves together is only useful for multiple flows that traverse the same network path; also, flow bit-time curves may feature non-linear properties that would complicate the literal mathematics of flow aggregation.

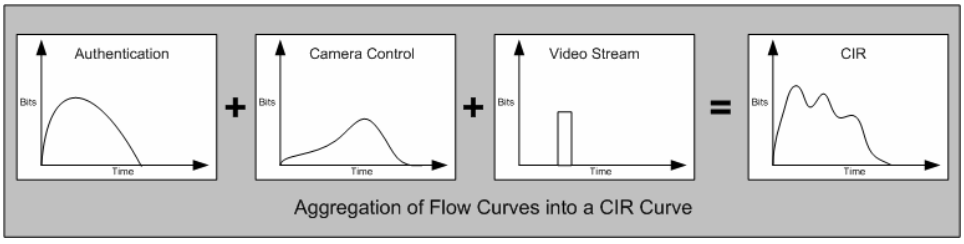

Figure 7. Aggregating Bit-Time Curves for Flows

Links can also be described by bit-time curves. In their case, the curve generally starts at the origin, since zero bits can be transmitted in zero time. Over the minimum time that it takes for bits to transit from source to destination, the value of this curve remains zero, after which it steps or slopes to a constant positive value corresponding to the average behavior of the link. This value is determined by throughput, latency, loss, and any other parameters that influence the effective rate of bit transfer. In general, increasing the throughput increases this value, whereas increasing the latency and loss decreases it. An example bit-time curve for a single link is shown in Figure 8.

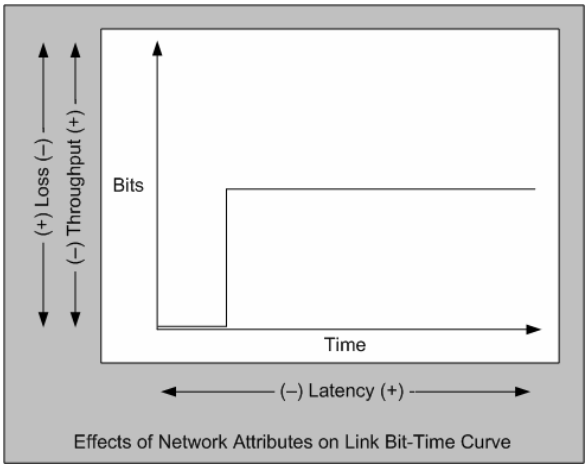

Figure 8. Example Bit-Time Curve for a Single Link

It is also possible to aggregate several link curves into a path curve, though the mathematical relationship would be different than that for flows. Combining link curves generally yields a smaller path curve; in part this is because the throughput of the path is no greater than the lowest throughput of any link. The curve will also be reduced because each link's latency and loss accumulates, further decreasing the number of bits the path can successfully transfer in a given amount of time. An example of bit-time curve aggregation for a path is shown in Figure 9.

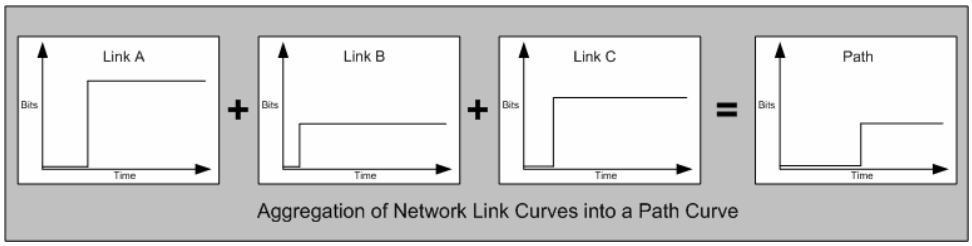

Figure 9. Aggregating Bit-Time Curves for Links

Yet another interesting feature of this model is the ease of comparing flow or service requirements against link or path capability. The area under a bit-time curve over the interval [0, t] represents the bits required or transferred, respectively, within time t. This means that a flow's requirements can be compared against a path's capability simply by overlaying the integral of one curve over the integral of the other, evaluated over the time *t* of interest. In other words, $\int_0^t B(t)dt$ where *B(t)* is a function that describes the

flow's bit-time curve and *t* is the time elapsed since the origin of the bits. This process is depicted in Figure 10. The highlighted region, below the flow curve and above the link curve, indicates bits that are *not* received before their value deadlines. If all bits can transit the network by their deadlines, there will be no highlighted regions.

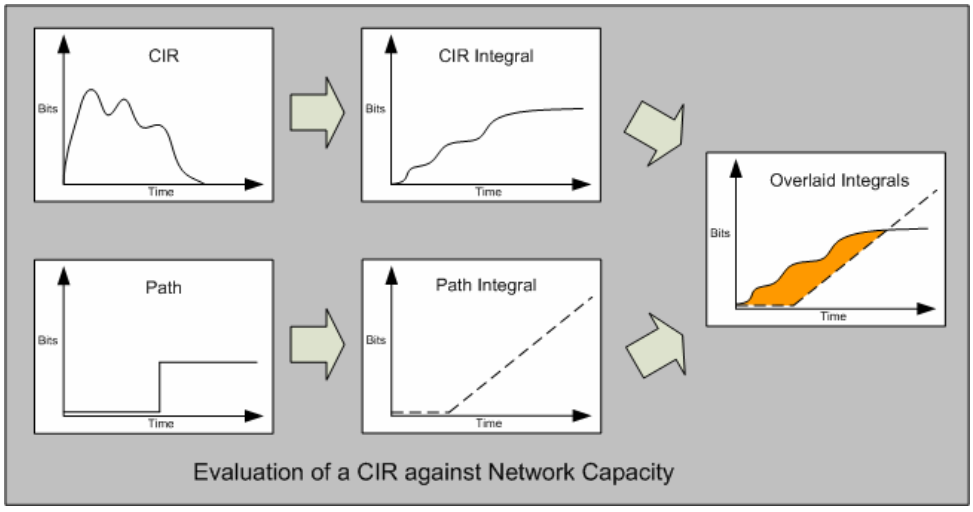

Figure 10. Comparison of CIR Requirements Against Path Capability

As discussed earlier, the relationship between the percentage of bits that arrive by their deadline and the "value" of a flow or service is complex and highly context-dependent. However, lacking a comprehensive analytical framework for each flow and service type, the highlighted regions can be thought of as detrimental to service value. The percentage of the total area under the flow curve that is not highlighted may be applied to a simple utility function specific to the service represented. This would provide a useful approximation of the value achieved by that service.

Non-linear flow and link properties alluded to above as well as the complex mapping between individual flows and paths complicates the mathematical relationships involved in aggregating flows and links and comparing services and paths. The simplified mathematics presented here only illustrate that this model allows for combining elemental components of the model into aggregates and comparing those aggregates, which in itself demonstrates the power of the model to describe network configuration and behavior. However, follow-on research must be performed to elucidate the exact

nature of these relationships in order to guarantee accurate assessments. Potential approaches to these challenges, including fuzzy logic and artificial neural networks, are discussed in the final chapter.

There is now a model for translating human-centric informational requirements into constituent flows, flow attributes, and finally bit-time curves that represent each flow's traffic in a value-deadline context. This model is capable of mapping flow requirements onto path capabilities to evaluate flow performance, and from this determine the achieved value of a service. A conceptual view of the translation and comparison processes is given in Figure 11.

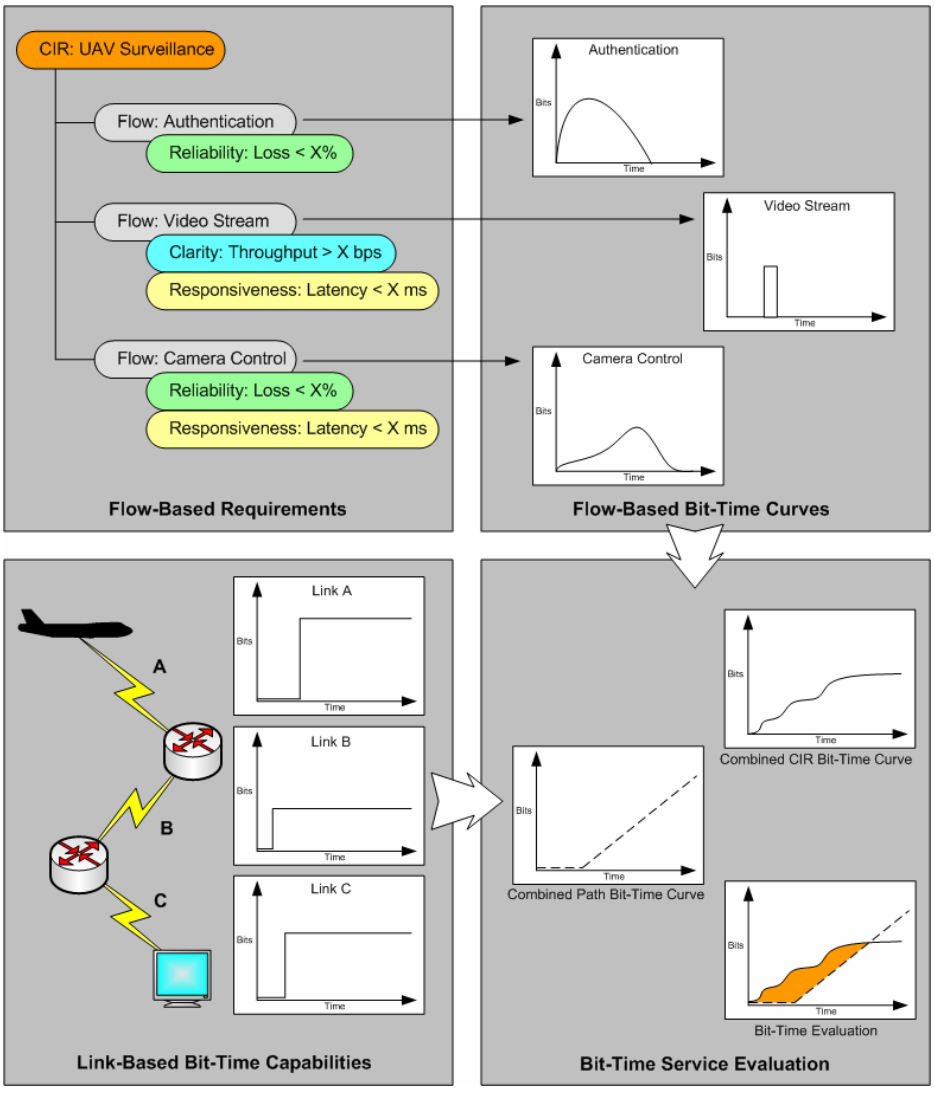

Figure 11. Evaluating a CIR Along a Specified Path Using the Bit-Time Method

Notice that there are two distinct sources of information feeding these processes. One comes from the user-specified CIR, which ultimately produces the bit-time curves for each flow. This process is discussed next. The other source, which generates the bit-time curves for each link, is the measurement architecture discussed in the following section. Since the specification of flow bit-time curves comes from a disparate process, the discussion will now turn to the specification of CIRs. Specifically, how a non-technical user is able to state information requirements in terms of services and flows and how those statements are translated into the inputs required by the proposed model.

4. Creating the CIR Language

Remaining is the specification of a CIR language based on the parameters elucidated from this discussion. These parameters are those shown in Figure 3 along with the essential properties of each constituent flow. The assembled model of a CIR is shown in Figure 12, and its components are described in the following paragraphs.

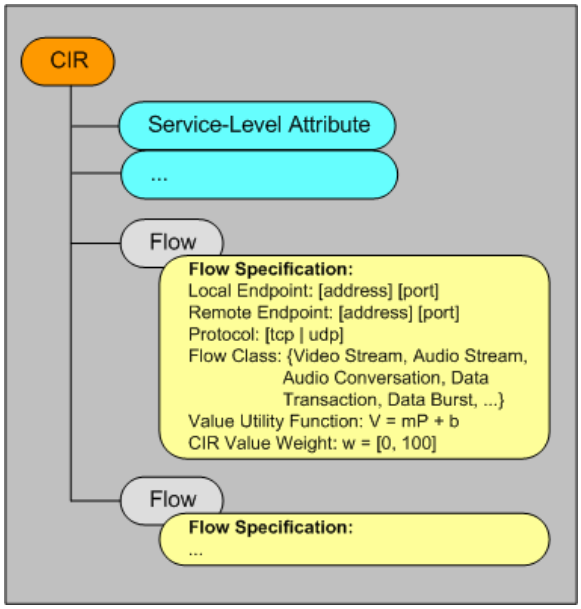

Figure 12. CIR Specification Model

At the top level of the model are general requirements for the CIR; these are the kinds of service-level requirements shown in Table 2. These requirements are the primary way the non-technical user can specify their needs without having a deep understanding

of the network traffic that comprises their CIR. Service-level requirements act as parameters that adjust the bit-time curves for each specified flow. For instance, a hypothetical web browsing service may feature a responsiveness requirement that a web page loads within 10 seconds. This 10-second parameter would scale the time axis of each flow's bit-time curve such that the value diminishes if all bits are not transferred within that timeframe.

Under the global service level is a description of each constituent flow. There are three essential components that must be specified for each flow: the path it traverses, its bit-time curve, and its relationship to the overall value of the service. For networks without multiple redundant paths between nodes, a flow's path can be derived from its endpoints. Network topology discovery mechanisms or human-inputted topological information can be used to determine the path. Flow curve descriptions may either be numerical parameters that define the shape of the curve or a classification of the flow type within a predefined set of curves. For the sake of this research, bit-time curves will come from defined classes of traffic based on empirically-derived flow characterizations. Service value relationships are defined by the combination of a utility function which relates the percentage of bits that arrive on-time to the percentage of flow value achieved, and a weighting factor that specifies the value of each flow to the overall service. For the purposes of this research, the utility function is represented by the linear function $V = m \times P + b$, where P is the percentage of bits that meet their deadline and V, the value to the service, is constrained within the range [0, 1] as shown in Figure 13. The variables m and b are calibrated for the particular flow type.

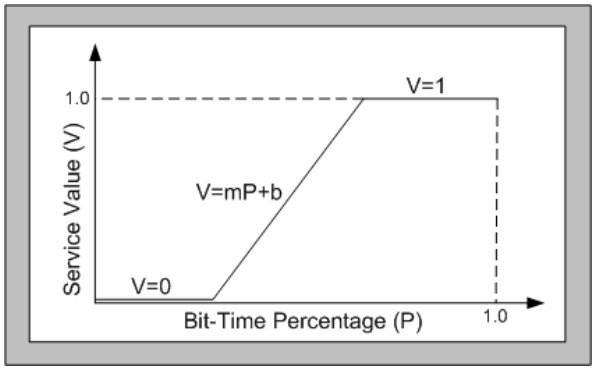

Figure 13. Utility Function Relating Bit-Time Evaluation to Level of Service

Taken together, these parameters form the fields of a data structure describing one atomic CIR. For illustrative purposes and as a baseline for testing conducted in the following chapter, a complete CIR language specification is defined here. There are technically two CIR data structures, one that represents the input required by the non-technical end-user, and another that fills in the requisite information to complete the model shown in Figure 12. Both data structures are presented in XML syntax for the sake of familiarity. These are presented in Figure 14, applied to the scenario used throughout this chapter.

User-Required Input

```
<CIR
        user_id="mrclement"
        cir_id="UAV_Surveillance_1"
        start="2007-08-12_17:30:00Z"
        stop="2007-08-12_18:15:00Z"
/>
        <Attribute
                type="Clarity"
                value="High"
        />
        <Attribute
                type="Reliability"
                value="Moderate"
        />
        <Attribute
                type="Responsiveness"
                value="High"
        />
        <Flows>
                <Flow
                        type="Streaming.Video.Surveillance"
                        remote="uav145.navy.mil"
                />
                <Flow
                        type="Burst.Data.Short_Control"
                        remote="uav145.navy.mil"
                />
        </Flows>
</CIR>
```

Completed CIR

```
<CIR
        user_id="mrclement"
        cir_id="UAV_Surveillance_1"
        start="2007-08-12_17:30:00Z"
        stop="2007-08-12_18:15:00Z"
/>
        <Attribute
                type="Clarity"
                value="75"
        />
        <Attribute
                type="Reliability"
                value="50"
        />
        <Attribute
                type="Responsiveness"
                value="75"
        />
        <Flows>
                <Flow
                        type="Transaction.Authentication"
                        local="192.168.99.123:1256"
                        remote="192.168.99.214:80"
                        value_m="1.3"
                        value_b="-2.1"
                        value_weight="85"
                />
                <Flow
                        type="Streaming.Video.Surveillance"
                        local="192.168.99.123:1423"
                        remote="192.168.99.214:1234"
                        value_m="1.0"
                        value_b="-3.6"
                        value_weight="50"
                />
                <Flow
                        type="Burst.Data.Short_Control"
                        local="192.168.99.123:1547"
                        remote="192.168.99.214:415"
                        value_m="1.8"
                        value_b="-1.5"
                        value_weight="65"
                />
        </Flows>
</CIR>
```

Figure 14. CIR Specification in XML

In the completed CIR specification on the right, all attributes from the CIR model are present. Service-level attributes are given by name and assigned some value; in this case, an integer on a scale from zero to 100, indicating the importance of that attribute. As noted earlier, these values may be used as scaling or weighting factors for the bit-time curves in individual flows. For each flow, the requisite information is provided. In this presentation of the model, bit-time curves have been pre-classified in their own taxonomy, and are referenced by name for each flow in the CIR. These types may assume a specific protocol, allowing a separate protocol field to be excluded. Start and stop times have been added so that an automated management system may know when CIRs will be in effect.

The user-required input to the specified CIR is more terse, containing only those pieces of information both critical to building the complete CIR specification and knowable to the non-technical user. Service-level attributes have been reduced to a user-specified "High," Moderate," or "Low," though if an appropriate user interface exists, numerical values may be derived from user selections. As discussed in the final chapter, it may be possible to present different service levels in a way that makes appropriate attribute selections more intuitive.

Flows are likewise simplified, specifying only the remote endpoint and the type of flow. Again, this alludes to the difference between what the user must know in order to articulate his or her requirements versus what must be known by the management system in order to effectively monitor the network. The endpoint and type of flow might be selected via an interface that allows the user to select service components from a list of assets and their corresponding capabilities. For instance, the asset named "uav145.navy.mil" would feature capabilities for surveillance-quality video and camera control. It may also feature other ISR and control capabilities that this user does not require. The interface would then translate the user's selections into specific endpoints, flow types, and parameters based on the input given. It would also fill in pre-requisite flows, such as authentication. If desired, the user may be able to make any final adjustments to the completed CIR before it is submitted to the management system.

Along with the CIR translation process that produces bit-time curves for each flow, this CIR specification process builds a complete procedure spanning from user selection of information requirements and service attributes through production of bit-time curves and configuration information essential to the management system. The two separate aspects of CIR specification and bit-time curve translation are also made explicit by this procedure. This is summarized in Figure 15.

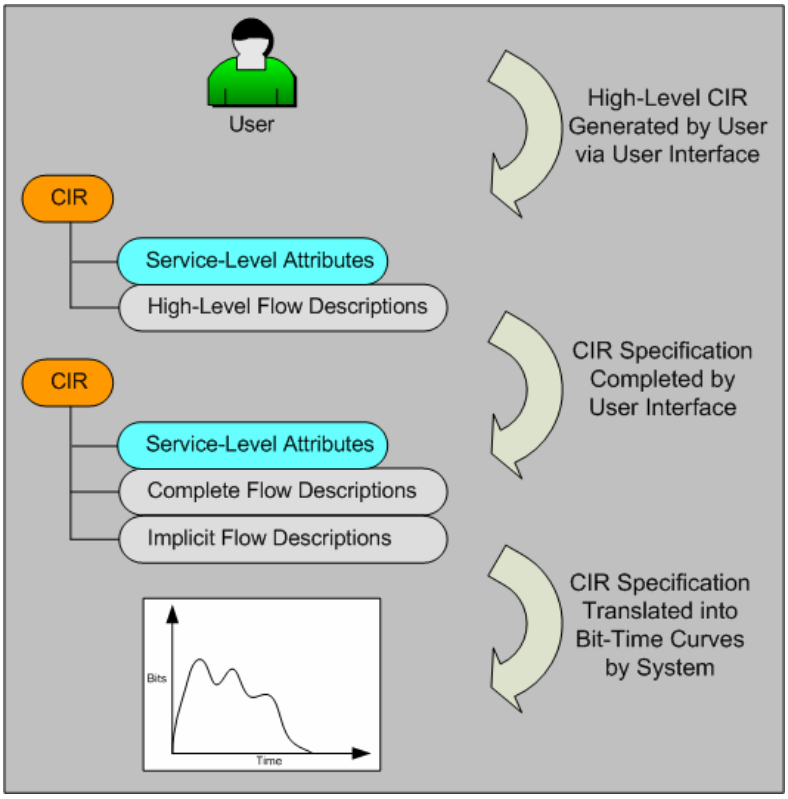

Figure 15. Specification and Translation of a CIR

The service specification and evaluation model presented here forms the crux of the holistic network management model proposed by this research. With the model established, it is appropriate to discuss how such a model would be implemented. The remainder of this chapter offers guidance on the architecture and implementation of both the underlying measurement infrastructure and the user interface and visualization capabilities necessary to making this model usable in an operational context.

C. MEASURING SERVICE PERFORMANCE

1. Measurement Infrastructure Considerations

Requirements for a holistic management model were described earlier in this chapter. Two of these requirements directly impact the design of measurement infrastructure: the management system must yield results in a useful period of time, and it must not interfere with operational traffic. Underlying both these directives and the broader technical requirements of such an architecture lay specific requirements which will be discussed here.

Most techniques for measuring the capability of a network link require generating and sending test traffic over the link, often in large bursts. The frequency of these assessments directly and adversely impacts the performance of operational traffic. Finding ways to utilize minimal amounts of test traffic, or better yet operational traffic itself, for capability measurement is important for the usability of this architecture. Ideas for implementing these techniques are presented later in this section.

The collection and processing of measurements is another area that can adversely affect operational traffic. If, for instance, the attributes of every link and every flow were transmitted across the network to one central management device once each second, constrained links could become overloaded by management traffic alone. Distributing collection to several topologically-scattered devices may mitigate these effects. Each device could gather measurement from network elements nearby, and summarize those measurements into the minimal information needed to be sent up to the next tier of the network. As shown in Figure 16, measurement nodes throughout the network could report to distributed management nodes, which in turn summarize measurement data and report to one or more centralized management nodes that collate and process management data from across the network.

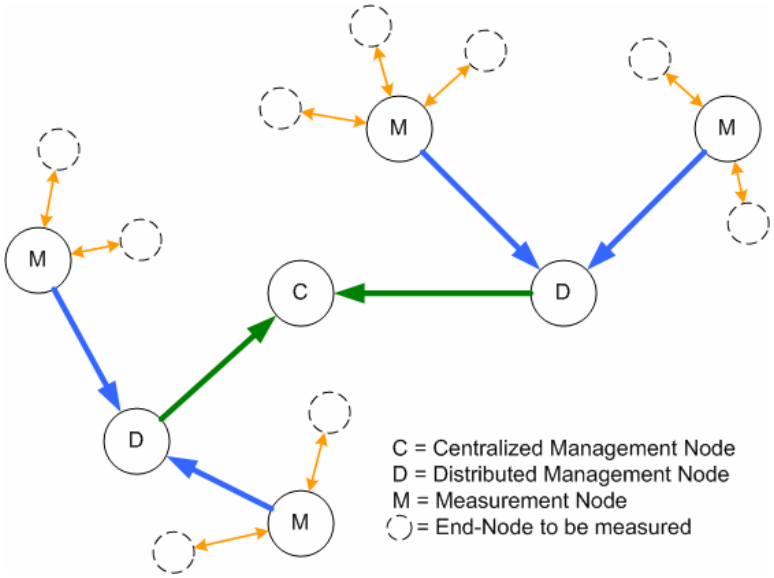

Figure 16. Topologically-Distributed Measurement Infrastructure

Time synchronization, though less obvious, is one of the most critical factors. Some of the techniques presented in this section require cooperative measurement between devices. For instance, packets may be captured near the source and near the destination of a flow, to determine factors such as latency and loss along that flow's path. In order to match packets between the two capture points, precise time synchronization is paramount.

These provide general guidelines for designing the measurement infrastructure. The following sections describe in more detail the conceptual process of characterizing both flows and links as bit-time curves. Examples are given to illustrate the basic methodology; however, the specific tools and techniques are left for discussion in Chapter Four. Finally, the method of comparing a flow or service against a path and evaluating the level of service achieved is discussed in more detail.

2. Characterizing Flows with Bit-Time Curves

The proposed model does not perform live analysis of flow traffic to determine the level of service achieved. Rather, it relies on knowledge of both the flow's requirements and the path's capabilities expressed as bit-time curves in order to perform a comparative analysis that determines the expected level of service. As such, it is

necessary to have characterizations of each flow type to use in these comparisons. Since most flow types correspond to well-established protocols, an effective approach to baselining would be to build a library of known flows and their bit-time characterizations. This would be an offline process, happening prior to or in parallel with but separate from live network management operations. These characterizations could then be used for evaluating levels of service in real time.

As discussed earlier, the bit-time curve for an individual flow represents a snapshot of that flow at any instant. At any given time, there should exist bits in transit from source to destination whose value deadlines statistically match the flow's bit-time curve. The empirical characterization method builds this curve based on observation of flow traffic in ideal network conditions. For some flow types, the traffic observed under these conditions matches exactly the requirements for perfect service. Video streams, for instance, have strict value deadlines for individual frames of video; a late frame is a lost frame. Other types may provide more leeway in the arrival times of individual bits. If a single text chat message takes extra time to arrive, it may be unclear if any value is lost. In these cases some additional analysis may be required to determine the boundaries for achieving perfect service.

Consider a simple video stream where each packet contains one video frame. Each packet is essentially a grouping of bits that arrives at the same time; loss, latency, and jitter could all be defined for packets rather than individual bits within a packet. Latency by itself is immaterial unless there are service-level attributes that mandate an upper bound on it; in plumber's terms, the length of a pipe does not matter as long as enough water continues to flow through it. Jitter and loss are more important, since these cause individual bits to not arrive "on time." Video receivers often employ a buffer to compensate for these effects. In the un-buffered case, all bits must arrive by a constant deadline in order to sustain smooth, clear video. The corresponding curve would have a single spike at that deadline. Conversely, buffered video can compensate for some percentage of bits that are late to arrive; the corresponding curve would change to represent one primary deadline, and a secondary later deadline based on the size of the buffer. Note that there are no lower bounds on the time at which bits must arrive. Bit-time

curves characterize the “no later than” time for each bit in transit; bits that arrive ahead of their deadlines have no negative consequences in this model.

One way to empirically characterize a flow is to compare its traffic at both the source and destination. From this, it is possible to build a distribution of bit arrival times, which for flow types with little leeway will closely match the bit-time curve required to achieve perfect service. For flow types that are close to streams, the resulting curve will likely appear as a spike as described above. For bursty flow types, the curves may be more interesting, depending on the variation in acceptable arrival times of individual messages.

Once a baseline exists for a flow type, simple parameterization based on service-level attributes may be added analytically. As discussed earlier, most service-level attributes “stretch” the bit-time curve along either the bits or the time axis. Assume the video stream described above uses frames 100 kilobits in size and has a deadline at 500 milliseconds in un-buffered mode; the bit-time curve will have a spike sized proportionally to the frame size and centered at that time. If the user wishes to increase clarity such that 200 kilobits per frame is used, the curve will stretch on the bits axis accordingly. Likewise, if the user imposes a bound of 75 milliseconds for responsiveness, the curve will contract on the time axis. However, in both cases, the basic shape of the curve will stay the same, since the essential relationships within the flow type remain unchanged.

3. Assessing Link and Path Capability

Similar to a flow’s Bit-Time curve, the curve for a link or path represents the number of bits that can transit from source to destination over time, starting from the moment when the bits are sent. Since there is always some latency, the value of this curve will be zero until the minimum time at which a single bit may have transited. Throughput, jitter, and loss also affect the shape of this curve.

On the macro scale, the capability of a link or path can be evaluated simply by sending traffic from source to destination at the maximum possible rate, and measuring the number of bits that successfully arrive over time. Such a test by its nature would

account for average loss, jitter, latency, and throughput. However, this approach suffers from two limitations. First, it does not account for variation over time in the number of bits that are successfully sent; jitter is one important variable that fluctuates on the micro scale. Second, such an assessment would require the entire link or path to be utilized solely by test traffic for the measurement period, which violates the requirement of minimally impacting operational traffic. Hence, it is important to consider assessment options that adequately capture the capability of the network without adversely affects its operation. As the proposed formulation of link bit-time curves does not separate each performance dimension, this study is left for future research; this section focuses on capability assessment techniques that provide adequate information without negatively impacting operational traffic.

(Prasad et al. 2003) discusses a number of techniques for assessing network capacity, or "bandwidth," assuming only the endpoints of a path may have measurement capabilities such as specialized software. They categorize measurements into three types: capacity, the overall capability of the link or path; available bandwidth, the capacity not utilized by current traffic; and bulk transfer capacity, the effective throughput of a TCP flow. It also distinguishes between techniques that yield overall path measurements and those with granularity down to the individual links along the path. Table 3 summarizes their results.

Metric	**Link**	**Path**
Capacity	Variable Packet Size	Packet Pair/Train Dispersion
Available Bandwidth	n/a	Self-Loading Periodic Streams Trains of Packet Pairs
Bulk Transfer Capacity	n/a	Emulated/Actual TCP

Table 3. Capacity Assessment Techniques for Links and Paths

Capacity is the metric of most interest to the proposed model, since the goal is to compare overall requirements of the network against overall capability of the network. However, capacity also comes with the caveat that most measurement techniques inhibit operational traffic from transiting the network at the same time as measurement traffic. Most techniques rely on precise timing of test traffic transiting from source to destination; operational traffic would interfere with these timings and skew the results.

Jacobson (1997) alludes to this difficulty, and accounts for it using a "min-filter" that removes variations in bit transit times. This technique does introduce some error; (Strauss et al. 2003) addresses these issues and presents a new tool called Spruce that is supposed to be both non-intrusive to operational traffic and more accurate. These tools are not assessed in this research, but the references provide insight for future implementers of this model.

For the sake of comparing flow requirements against path capabilities, it would be possible to simply evaluate each path that bears one or more flows. However, as discussed in the next section, there is benefit to knowing network capabilities to the granularity of the individual link. A simple solution would be to use path measurement techniques for every link *and* every path; however, a more elegant and holistic solution is to periodically measure each link, and use the algebra of bit-time curves to build path assessments. If a path consists of three links, each with its own bit-time curve, those curves can be aggregated into a single bit-time curve as shown in Figure 9. There are two main attributes to these curves: the time t at which the first bit arrives, and the constant bit arrival rate. The former is determined by the latency of each link, and can be aggregated as $L_{Path} = \sum_{l \in Links} L_l$ where L represents the minimum possible latency for a given link or path. This means the time t at which the first bit transits a path is the sum of those times for each link, which is represented in the bit-time curve by an apparent shift to the right.

On the other hand, factors such as throughput, jitter, and loss aggregate in a reductive way. Each subsequent link in a path will at best have no effect on the constant arrival rate, since path throughput is at best the minimum of all link throughputs and link losses accumulate into path loss. Jitter is an additional stochastic latency on top of the constant minimum latency present on a link, so it too degrades the arrival rate of bits. The exact mathematical relationships between these attributes are complex and will be left for future study; the tests conducted in the next chapter utilize tools to estimate the bit-time curves for links and paths.

4. Evaluating the Level of Service

As was discussed earlier, the level of service achieved is a function of both the requirements of the service and the capability of the path or paths supporting that service. This boils down to a comparison of each flow against the path carrying that flow, weighted by some utility function that defines the value of that flow to the overall service. As shown in Figure 10, the comparison involves integrating both curves and aligning them in time, such that the total bits required over the period [0, *t*] can be compared to the total bits transited over that same period. The comparative area under these curves reveals the performance of a given flow across a given path.

Bearing in mind that both curves represent relative snapshots in time, the values at time *t* show for bits transmitted at any instant, how many are due at time *t* versus how many should arrive by time *t*. This raises the question of how far along the time axis the two curves should be evaluated. Figure 10 shows a case where at first the requirements of the service exceed the capability of the path, but over time the path catches up and exceeds the demand of the service. If the level of service is simply the ratio of highlighted area to total area under the service curve, the evaluation period chosen will impact the result. Without extensive experimentation, it is difficult to say exactly how far the curve should be drawn, but intuition says that it should go as far as the latest-arriving bit in the flow's bit-time curve. This will produce a comparison of the complete behavior of the flow relative to the path carrying it. However, if the curve describes a bursty or stochastic flow where some bits are required in milliseconds and other are required in minutes, it may be necessary to limit the range that is evaluated to the point where majority percent of the flow's bits are included. This should be part of further study on this model.

Another interesting design question is whether to perform service evaluations strictly between flows and their corresponding paths, or to evaluate flows against each link within the corresponding path. Although path comparisons are useful for overall service evaluation, a holistic model should also be able to pinpoint specific bottlenecks and identify the relevant flows and links. This is what gives the network user an

advantage over traditional tools that report only aggregate performance metrics. To do this, it is necessary to know the capacity of each link, and to map each flow onto its corresponding links. This process is illustrated in Figure 17.

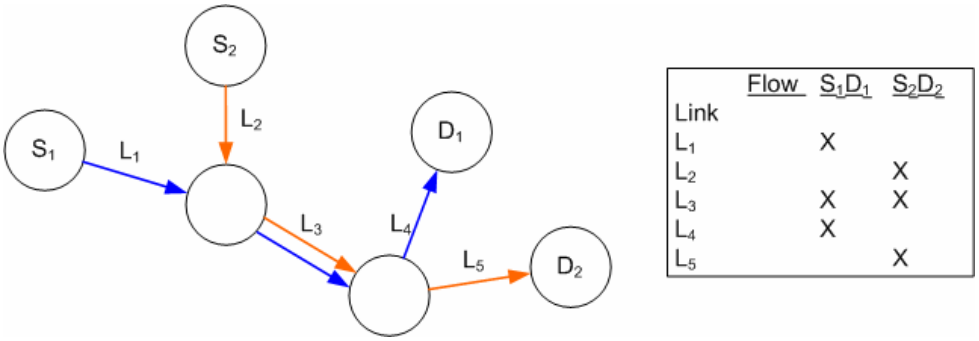

Figure 17. Mapping Flows onto Corresponding Links

Two flows, S_1D_1 and S_2D_2, transit paths between source S_1 and destination D_1, and S_2 and D_2, respectively. The table shows the mapping of each flow onto the corresponding links L_i in its path. Notice that both flows transit L_3, indicating that an assessment of L_3's utilization requires aggregating both flows and comparing that against L_3's current capacity. If the capacity is insufficient to fully support both flows, not only will the levels of service for each flow be calculated correctly, it is possible to systematically pinpoint both the specific links degrading service performance and the particular flows being degraded. With this information, network users can take action to correct performance issues, either by adjusting the configuration of the network or by reducing their own usage, self-prioritizing their traffic in order to achieve the highest value possible in the given conditions.

D. VISUALIZING THE NETWORK OF SERVICES

The final though arguably most important piece of the proposed model is the human interface. Once all CIRs are translated, every flow is mapped onto its path and links, and all measurements are taken, the resulting analysis must be presented to some decision maker in order for any of this process to be useful. The best representation of service performance in the world, if not presented in a meaningful way, is of little use for effective network management. Although this research does not directly investigate the

effectiveness of service visualization, it is important to address this topic as it significantly impacts the usability of this model. This final section describes some of the challenges in visualizing a holistic network management model, and presents some thoughts on how to work toward a usable solution.

1. The Problem of Holistic Visualization

Many approaches exist for visualizing the performance of a network. Some tools use red-light green-light iconography to indicate whether or not a node or link is available. Others use gauges or graphs to indicate the amount of traffic transiting a link at any given moment. Yet others present matrices of source and destination hosts or ports to represent the patterns of usage on the network. However, very few approaches exist to present several network metrics simultaneously, let alone alongside representations of flow- and service-level performance.

One of the difficulties is that each level has its own distinct set of attributes and a unique topology. At the network level, every computer, switch, and router is an element in the graph; each of these has performance attributes that affect the overall performance of the network. At the flow level, most devices fade from the picture, leaving only the endpoints of specific flows of traffic. Just mapping these two layers together can be a challenge; adding the services level is an onerous task. As shown in Figure 18, every layer has a distinct structure that can be difficult to represent in a single view.

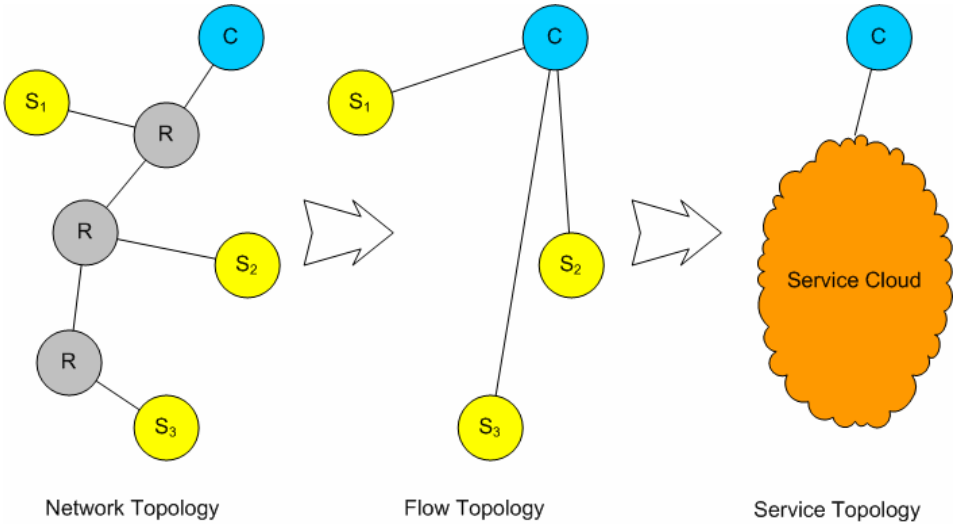

Figure 18. Network Topology at Each Layer

Even within a single level, representing all attributes is a challenge. Just three common attributes such as throughput, latency, and loss would be a challenge to represent intuitively on top of the network topology depicted above. Perhaps the thickness of each line could represent throughput, and its length could represent latency. Loss is a difficult attribute to represent; it might be portrayed by dashed lines versus dotted lines, or by line color if that was not being used for something else. Already the obvious physical characteristics are taken, and flow and service performance is not even considered. It is clear that another direction must be taken to visualize this information.

2. Promising Approaches

Although no tools yet exist to holistically visualize a network of services, there are some noteworthy approaches already taken. A brief discussion of these may be insightful toward the design of future visualization techniques.

Etherape (Ghetta and Toledo) is an application- or flow-level tool for visualizing the amount of traffic transiting between endpoints on a network. It presents all devices, listed by IP address, in a ring, and portrays each flow as a cone with the top at the source and the base at the destination. This gives an immediate and intuitive notion of which devices are sending and receiving the most traffic. The cones are color-coded to represent different protocols or flow types. Figure 19 shows Etherape running on a network.

Although Etherape is only designed for a single local area network, the architecture could be expanded using distributed measurement as described earlier to accommodate a larger internetwork. However, in its current design, it is difficult to align the visual placement of endpoints with the network topology. It also lacks a way to aggregate individual flow behavior into composite service behavior, and has no notion of link or path performance.

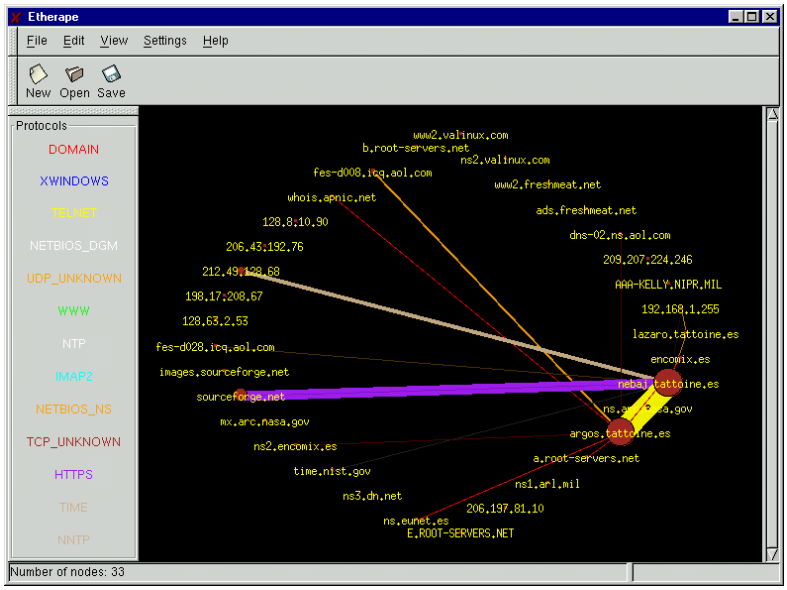

Figure 19. Screenshot of Etherape (From: Ghetta and Toledo)

Big Brother (Network Uptime) is one of the matrix-style visualizations as mentioned above. One axis is the set of devices on the network; the other is the set of services supported by those devices, such as HTTP, FTP, and SMTP. The values within the matrix are color-coded symbols indicating the availability of each service on each device: whether or not it is supported, and if it is, whether or not it is accessible at that moment. Figure 20 shows Big Brother in action. This visualization by itself does not contribute to holistic performance monitoring; however, if these axes were changed to CIRs and their constituent flows, or flows and their corresponding links, then a color-coded landscape indicating performance may yield useful information about the overall state of the service network.

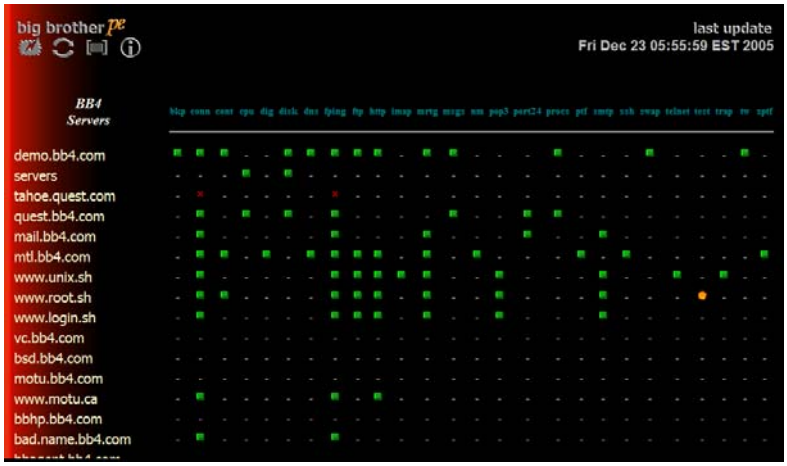

Figure 20. Screenshot of Big Brother (From: Network Uptime)

Otter (Ma) is the most topologically-focused tool of the three. It combines a topological view with visual placement based on specified criteria. Figure 21 shows an example for latency along the paths to several devices. Using visual placement along one axis as a means of representing data is a novel approach to depicting performance data while maintaining visual information about the structure of the network. Color-coding of individual nodes provides an additional dimension of representation. Notice that the alignment of nodes on the vertical axis obfuscates the topology in a few places; it is important to consider the effects that each layer of representation will have on the clarity of every other layer.

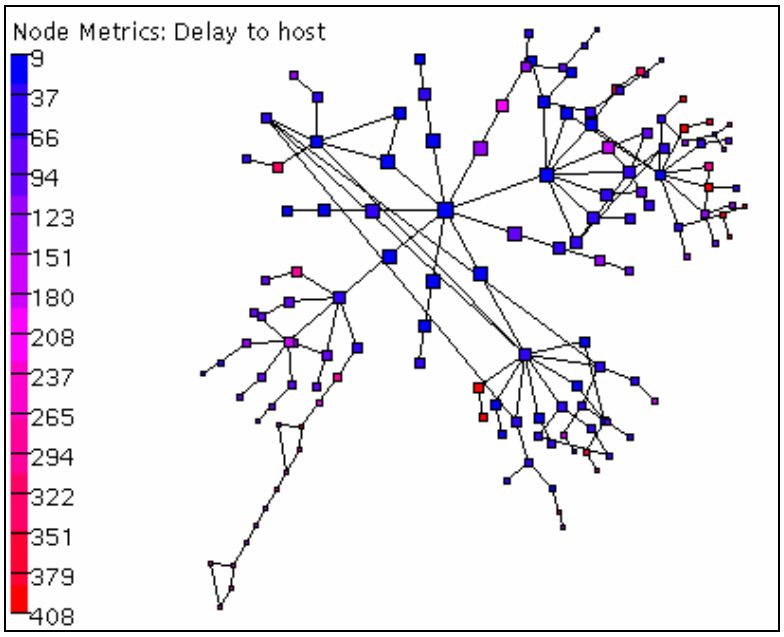

Figure 21. Screenshot of Otter (From: Ma)

3. Future Directions

A holistic network visualization tool must be simultaneously informative to the network user who wants to evaluate the level of service achieved for each of his or her CIRs and informative to the network administrator who wants to monitor the network and determine areas that require configuration changes or resource expansion in order to accommodate users' needs. It must start with the highest level and report the achieved quality of every CIR active on the network, and be able to drill down into each flow, path, and link to provide timely and concise information about the performance and behavior of the network.

This will likely require a layered approach, where different dimensions of information may be turned on or off as desired by the user. It is possible that a generic visualization engine that allows users to create their own customized views, mapping network and service data onto dimensions of placement, size, shape, and color, would be beneficial. Within the military domain, geographical information should be available, enabling the infrastructure to be overlaid on a map; this is especially useful in tactical wireless environments where devices are distributed across larger areas than would be the

case in a single building or facility. Finally, the user must at all times be able to quickly ascertain the state of the network, without sifting through disproportionate quantities of extraneous information. A minimalist approach to user interface elements such as menus and the ability to access critical information quickly should be driving design goals.

Relating these requirements back to the Bits in Time model proposed earlier, the aggregation and comparison algebra of bit-time curves should be reflected in the visualization mechanism. This means both that services and flows must be depicted along with some graphical representation of their relative network requirements, and that links and paths must be presented in a way that denotes their relative capacities. Since the proposed model enables component requirements to be aggregated into composites of services and paths, it is possible that an appropriate visualization would allow the network topology to be expanded and collapsed along the same dimensions. A single node, representing a service and color-coded or otherwise marked to denote the current level of service achieved, could be expanded into a topology of vertices representing endpoints and edges representing flows along their distinct paths. These in turn could be marked similarly to the single service-node in such a way that intuitively presents the status of each service and each link. In order to achieve the drill-down capability that enables pinpointing network issues, each level could be expanded, breaking the composite service level shown at one level into the component service levels of each piece.

Although stated as the most important aspect of this model, visualization is also the most briefly treated. This is in part due to the difficult nature of the problem: the multiple facets that must simultaneously be displayed, the requisite intuitiveness of the presentation, and the necessary ease of access to information that drives critical network management decisions. Rather than present an ill-formed solution within this research, the development of an appropriate visualization is left for future research and experimentation. The following chapter focuses on the specification and translation model presented in this chapter, and a preliminary discussion of the specific measurements necessary for service performance evaluation.

IV. TESTING AND RESULTS

A. OVERVIEW

1. Testing Goals

The previous chapter laid out the requirements for and components of a proposed holistic network management model. It is necessary to test this model in order to validate its accuracy and viability. The model's accuracy is tested to determine if the model correctly expresses the expected phenomena; specifically whether it satisfies the functional requirements defined in the previous chapter. Likewise, viability is tested to determine if the model is feasible to implement and utilize in real-world operations. This chapter documents preliminary testing of the model, and proposes changes to the original model based on lessons learned.

These goals stated, only a subset of this model can feasibly be tested within the scope of this research. Many of the collection and analysis processes are simplified or performed manually in order to demonstrate the viability of the model and to assess its accuracy. This initial testing leads to some preliminary conclusions about and revisions to the proposed model, which are presented throughout this chapter and in the next.

2. Testing Environments

The tests performed on this model are divided into two parts: lab tests and field tests. These correspond with the testing goals of validating accuracy and viability, respectively. Lab tests are set up to demonstrate the concepts of the model in simplified scenarios, in order to test individual aspects of the model. Field tests follow to assess the usability and utility of the model in a live scenario simulating an operational environment. The following sections describe the configurations of these environments in detail.

B. CENETIX, TNT, AND MIO

Testing was conducted utilizing facilities and experiments run by the Center for Network Innovation and Experimentation (CENETIX) at the Naval Postgraduate School, which is directed by Dr. Alex Bordetsky. Established in 2004, CENETIX conducts research into emerging network technologies as well as models of networking and collaboration. CENETIX supports two field experimentation programs on a quarterly basis: the Tactical Network Topology (TNT) at Camp Roberts, California, which is conducted in cooperation with U.S. Special Operations Command (USSOCOM) and directed by Dr. David Netzer at the Naval Postgraduate School; and the Maritime Interdiction Operation (MIO) experiment in San Francisco Bay, which is conducted jointly with Lawrence Livermore National Labs as well as other joint and coalition defense agencies. The MIO program is led by Dr. Bordetsky.

C. LAB TESTING: CHARACTERIZING FLOWS AND LINKS

1. Test Environment

The initial lab test environment was as shown in Figure 22. It consisted of data sources, in this case video streams; one data receiver; a network link emulator; and traffic monitoring systems. It also included a time server for synchronizing data capture on the monitoring systems; this was to enable easy comparison of traffic near the sender versus the receiver. The primary flow of traffic is designated in the figure with bold arrows.

For the first test, the goal was to assess a basic data flow traversing a simple network. Streaming video was chosen for the flow type since it uses a single socket connection and should have a simple bit-time curve, as opposed to complex, bursty, or stochastic flow types such as web browsing and text chat. Two video sources were to be used: a Pelco (Pelco Corporation) network video server attached to a live camera and a pre-recorded video served via VideoLAN media server software (VLC Team). Pelco video is accessible via a webpage, and is available either as MPEG-4 or an MJPEG-like "Server Push" mode. The pre-recorded video is accessible via the VideoLAN media player, and is available in several streaming video formats.

The network link emulator was introduced to create artificial constraints on network resources. This would enable testing the performance of the video under varying conditions, such as with increased latency, jitter, and loss, or decreased throughput. These tests used the NIST Net (NIST Net) network emulation software, running on a small form-factor Linux device. The device was set to act as a router, enabling network effects to be applied to traffic passing between the subnets on either side of the device. This device also featured a third network connection, used solely for managing the device and configuring the emulation settings.

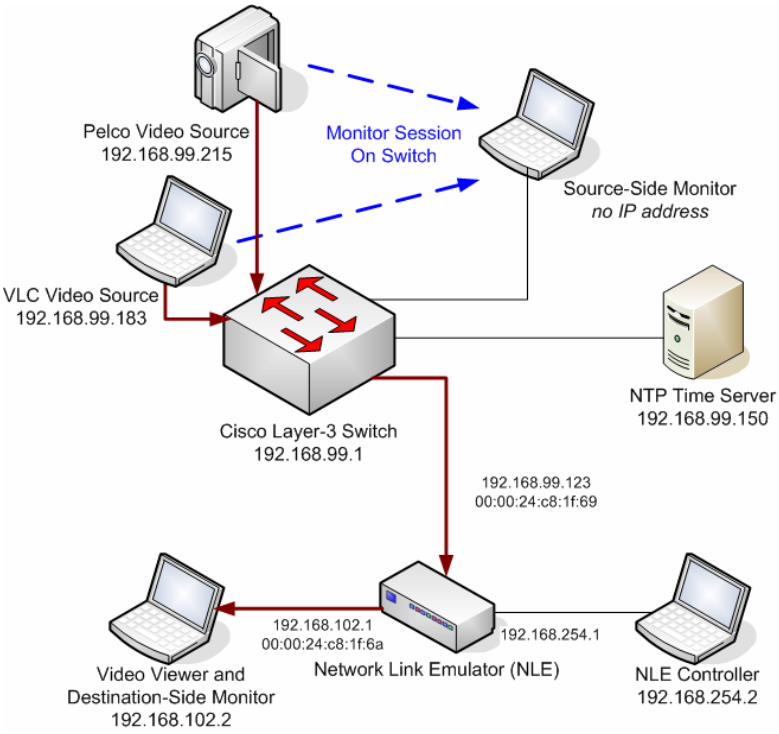

Figure 22. Initial Lab Test Environment

Traffic monitoring was performed using Wireshark (Combs). This enabled full traffic capture for subsequent quantitative analysis, as described in the following sections.

2. Characterizing Video Flows

The first test was performed with the Pelco video server using "Server Push" mode over a TCP connection. Early traffic analysis revealed that this mode used continuous HTTP GET requests initiated by the receiver for a JPEG still image resident

on the server; the effective frame rate of the video was bounded by the rate at which the client could execute HTTP transactions against the server. This led to very low frame rates, less than one frame every three seconds in some cases, which was not an acceptable baseline for performance comparison. It was also realized that the cycle of HTTP transactions would create a more complex flow to characterize than an actual video stream.

At the same time, another problem was discovered. The time synchronization relied upon to assist in aligning sender's and receiver's traffic captures was not providing sufficient precision to accurately align the data. Every captured packet is timestamped; these times were to be used to determine the time between each packet's origin and its arrival, thus establishing the effective bit-time curve for that flow under given network conditions. Data collected from the first test showed packets arriving at their destination *before* they were sent. Subsequent repeat tests and alterations to the time synchronization configuration did not yield any improvements. It became necessary to determine an alternative method of aligning packets in time.

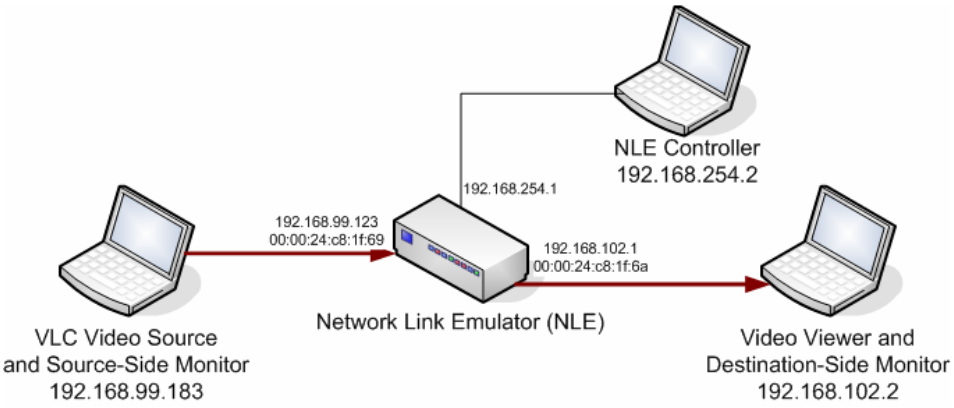

Figure 23. Simplified Lab Test Environment

Given these difficulties, the network topology was further simplified and the VideoLAN video server replaced the Pelco device. The simplified network is shown in Figure 23. MJPEG video over a UDP connection was chosen for the flow as this maintains an approximately constant bit-rate. Also, because the video does not use temporal compression and because UDP offers no traffic control, data loss at any moment

should not affect subsequent frames, and the sender will continue to transmit at a given bit-rate throughout the stream. Finally, the client was set up to not buffer video before playback; this would intensify the effects of adverse network conditions.

To form a baseline characterization, certain network conditions were chosen to represent the ideal case. The emulator was configured to add 50 milliseconds of latency to all traffic in both directions, but not to constrain the network in terms of throughput, jitter, or loss. This is comparable to a lightly- to moderately-loaded wired internetwork, and is on the same order of latency as traffic traversing long distances across the Internet. The effective latency between the two endpoints was verified using the ping utility, as shown in Figure 24. An average round-trip time of 100.7 milliseconds corresponds with 50 millisecond latency in each direction plus minute processing and queuing delays introduced by each device through which the packet passes.

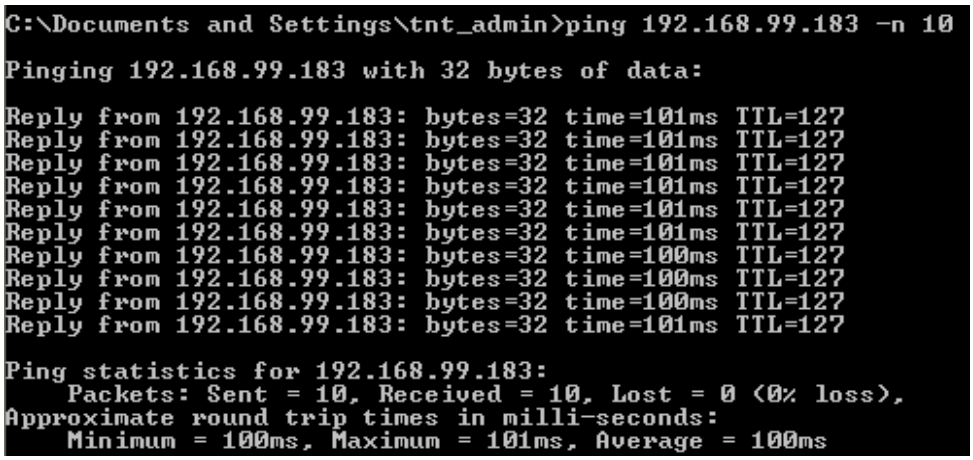

Figure 24. Round-Trip Time as Determined by the Ping Utility

The premise that enables this baseline characterization is that, for un-buffered video, there is no leeway for adverse network effects if perfect service is to be achieved. Without a buffer, any jitter, loss, or other effect that inhibits packets from arriving at a constant rate and in the correct order will cause noticeable defects in video quality. In other words, the bit-time distribution observed when the video is at peak quality is exactly the bit-time curve *required* to achieve that quality. To find this curve, traffic captures were taken on both the source and destination devices, and those captures were compared statistically to find the distribution of packet transit times. Using traffic

captures exported from Wireshark and a simple text-processing script, the transit time for each packet was tabulated as shown in Table 4.

Source Time	Checksum	Destination Time	Checksum	Absolute Delta	Adjusted Delta
21:01.361	0x4ecb	21:00.739	0x4ecb	00:00.622	00:00.051
21:01.365	0xf60d	21:00.742	0xf60d	00:00.623	00:00.050
21:01.369	0x23c7	21:00.746	0x23c7	00:00.623	00:00.050
21:01.373	0x7aaf	21:00.751	0x7aaf	00:00.622	00:00.051
21:01.376	0xc577	21:00.753	0xc577	00:00.623	00:00.050

Table 4. Sample Packet Transit Times from Baseline Test

The source and destination times are the absolute times reported by the respective traffic captures. Notice that the destination times predate the source times, giving rise to the synchronization problems discussed earlier. Checksums are codes used in packets to verify the integrity of the packet from source to destination; they are based on the content of the packet, so it is unlikely for two consecutive packets to have the same checksum. These were used to align the messages from the source capture with those from the destination capture. The absolute delta column shows the absolute value of the difference between the source and destination. Since these are known to be inaccurate, it was necessary to devise a method of adjusting the deltas. The adjusted delta column uses the average absolute delta and the observed round-trip time from Figure 23 to calculate the adjusted delta. The formula for this is $D_{Adj} = T_{Dst} - T_{Src} + \overline{D_{Abs}} + \frac{RTT}{2}$. As shown in Table 4, the adjusted delta values are on the order of the expected 50 milliseconds, though individual packet variations are preserved. The distribution of packet transit times, zoomed into the peak region, is shown in Figure 25.

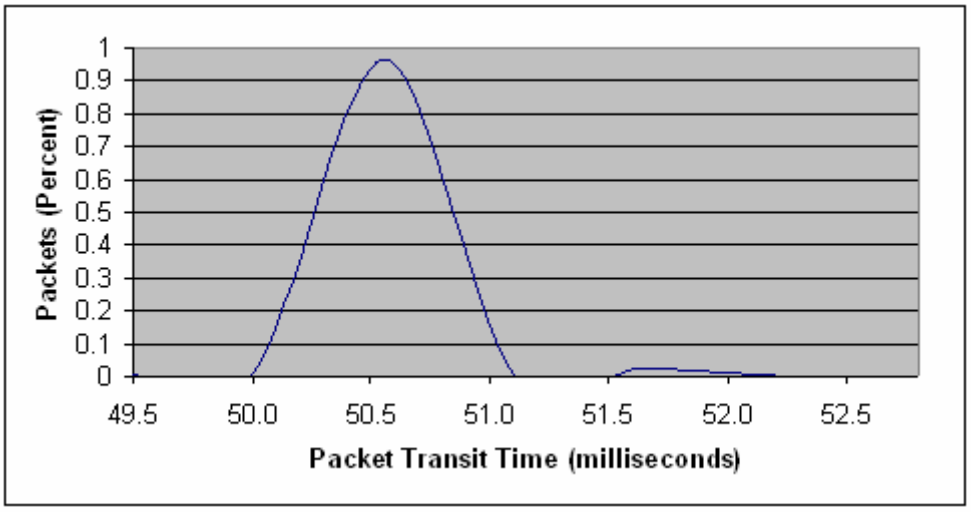

Figure 25. Distribution of Packet Transit Times

As predicted in the discussion from the previous chapter, un-buffered video in a low-jitter network forms a tight spike around the average latency of the path; in this case, just over 50 milliseconds. Interestingly, there is a small secondary bump just more than one millisecond later. Even without a buffer, video performance was without noticeable blemish even with some packets arriving slightly later than the bulk of the flow.

It turns out that this curve should have the same shape as the bit-time curve for this flow. In order to correctly express the bit-time requirement of the video, the area under the curve by any arbitrary value deadline must equal the number of bits that must transit within that period of time; this adjustment simply requires multiplying by a scalar that represents the bit-rate of the flow over that time period. To determine this quantity, the average number of bits arriving over the period from the earliest arrival to the latest arrival on the curve in Figure 25 is determined from captured traffic. The area under the curve is then adjusted to match the result. From the traffic capture, the average bit rate is known to be 1,408,070 bits per second. If this amount of traffic were spread over millisecond intervals, it would be 1,408 bits per millisecond. Applying that rate over the arrival time distribution produces the curve shown in Figure 26.

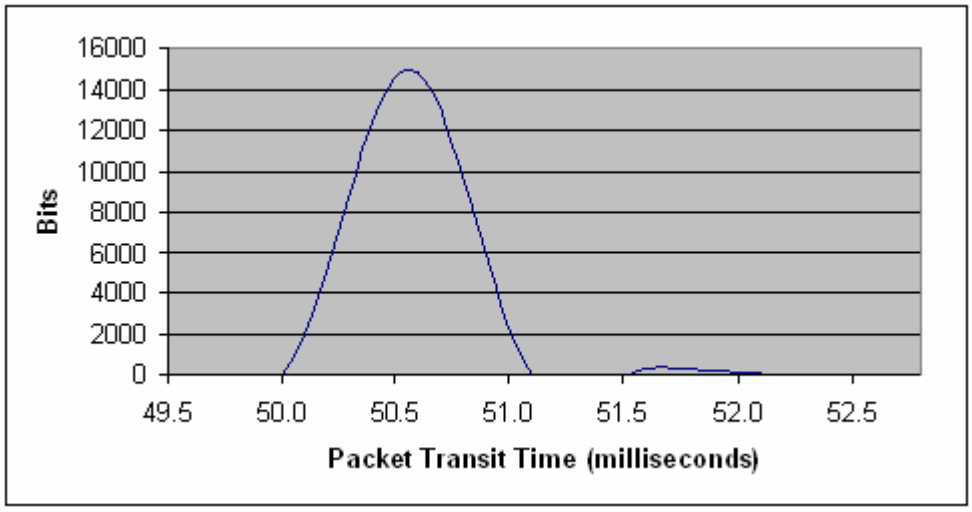

Figure 26. Bit-Time Curve for MJPEG Video Stream in Lab Test

3. Characterizing Link Capacity

The factors that affect a bit-time curve representing link capacity are the minimal or average link latency and the effective throughput of the path over time. Latency determines the time *t* at which the first bit arrives, and throughput determines the value of the curve on the bits axis after time *t*. These factors are illustrated in Figure 27.

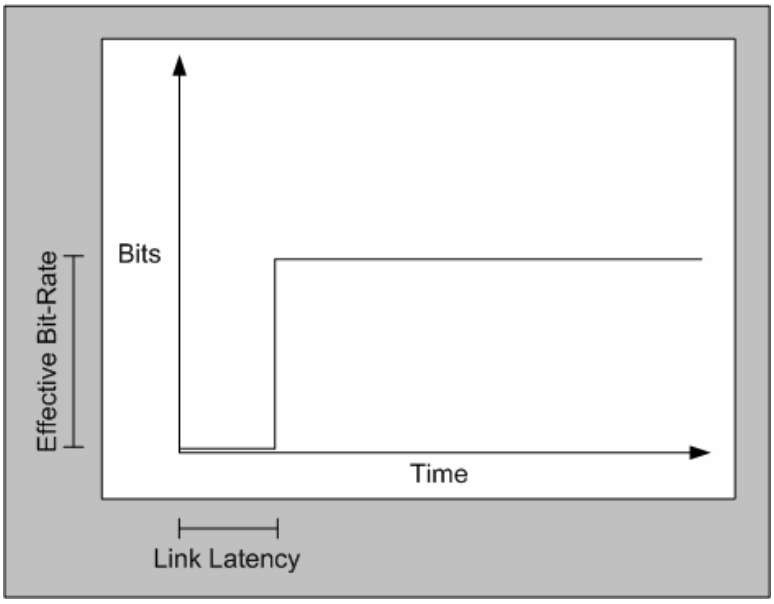

Figure 27. Mapping Network Factors onto Link Bit-Time Curve

Any appropriate network capability measurement technique can be used to find this curve. In-depth discussion of measurement methodologies is presented in the previous chapter; for this testing, a simple emulated traffic test using Iperf (Tirumala et al.) was used to determine capability. Iperf sends a specified rate of traffic from a given source to destination for some period, evaluating the effective throughput, jitter, and loss of that transfer. A simple ping can be used in addition to determine link latency. For the sake of simplicity, Iperf was run across the entire path rather than for each link. The results are shown in Figure 28.

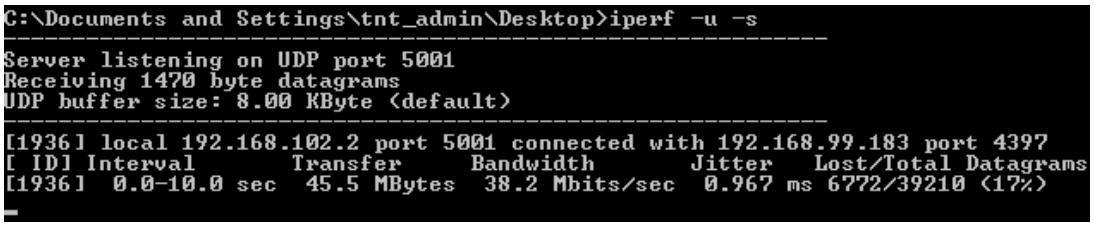

Figure 28. Results of Iperf Path Measurement with 100 Megabit-per-Second Traffic

The effective throughput, or "bandwidth," shown takes jitter and loss into account; otherwise it would be necessary to analytically account for those effects. It is also important to note that for these tests the attempted bit-rate was 100 Megabits per second, which is the theoretical maximum speed of the Fast Ethernet links used; some loss is expected as this exceeds the actual speed of the path. From the results shown in Figure 24, the latency is known to be approximately 50 milliseconds. In order to represent the bit-time curve, the effective throughput in bits per second must be spread over the area under the curve for that time period; this is given by $T_{Interval} = \frac{T_{bps}}{\Delta t}$. It is important that Δt is identical for the flow and the link or path. For this case, the granularity is in milliseconds, so $\Delta t = 0.001$. This gives the path bit-time curve shown in Figure 29.

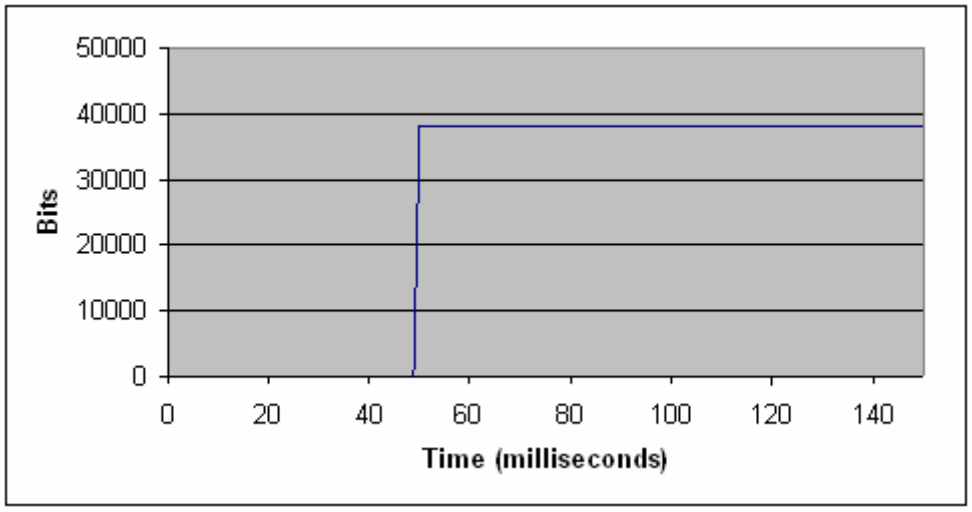

Figure 29. Bit-Time Curve for Path in Lab Test

4. Accuracy of Service Description and Prediction

If the bit-time curves depicted in Figures 26 and 29 were compared using the method described in the previous chapter, the path curve should exceed the flow curve, since the performance of the flow was considered perfect. This is depicted in Figure 30.

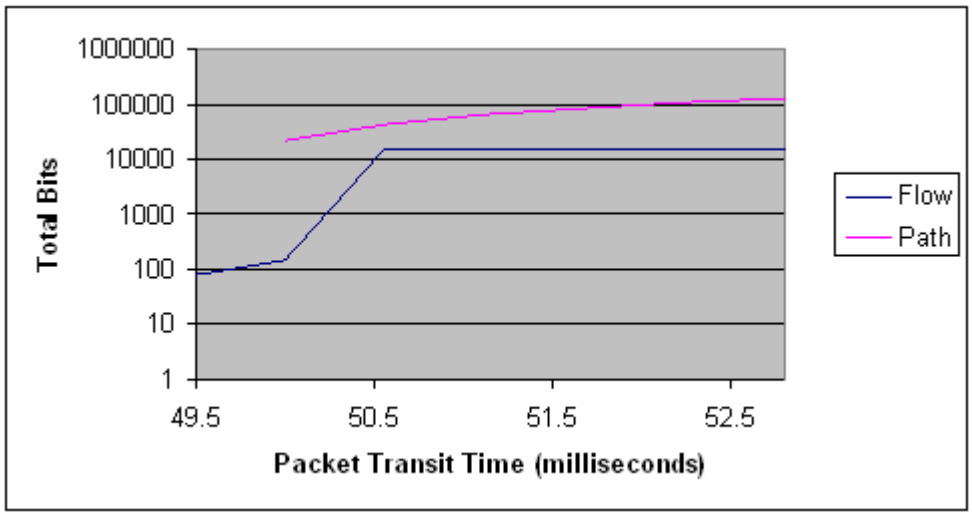

Figure 30. Comparison of Flow and Path Bit-Time Curves

Notice that the scale of this graph is logarithmic; the path bit-time curve exceeds the flow bit-time curve by at least half an order of magnitude at any time. This

corresponds with both the observed performance of the video and the captured data. The data indicated that the video required an average of 1.4 Megabits per second versus the measured path throughput of 38.2 Megabits per second. This appears to demonstrate that the model accurately describes both flows and links, providing valid results for a known case.

The other necessary test is of the model's predictive capabilities. For an altered set of circumstances, such as a diminished path capability, the model should predict a drop in service level. Two cases were tested: first, the path latency was increased from 50 milliseconds to 75 milliseconds; second, a five-second jitter was added to the 50-millisecond constant latency. In both cases, the video was observed and perceived quality is known. The test is of how well the model predicts the relative service levels in these cases.

In the case where the latency was increased to 75 milliseconds, the effective throughput should have remained the same, but the path bit-time curve should effectively shift to the right due to the greater path latency. The observed throughput from an Iperf was 38.1 Megabits per second, approximately the same as the first test. Figure 31 shows this bit-time curve.

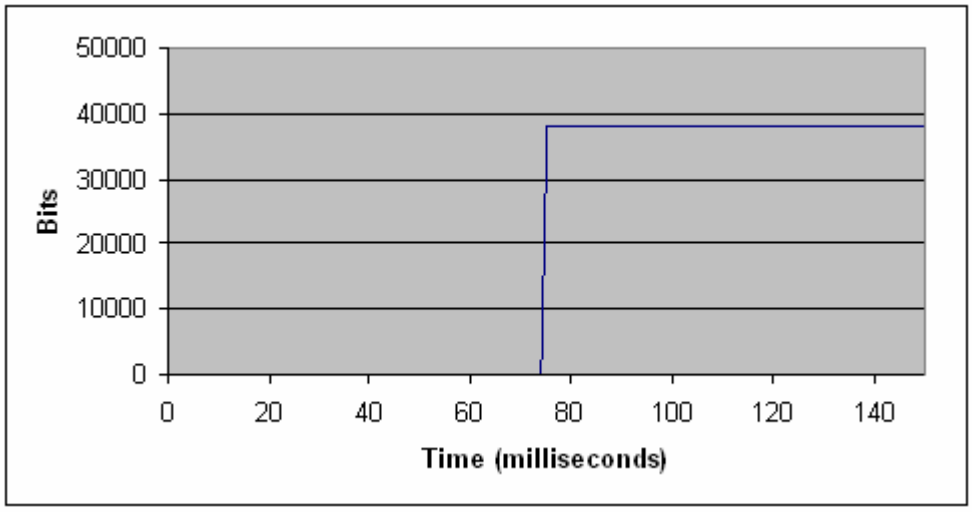

Figure 31. Path Bit-Time Curve for 75 Millisecond Latency

Comparing this curve with the flow's bit-time curve revealed the first discrepancy in the model. If the two curves were directly overlaid as shown in Figure 32, it would appear that there is a large deficiency in service performance with the link curve lagging far behind the flow curve; however, the observed video was of high quality, albeit with a slightly higher latency. This elucidates the point that some deficiencies determined by the bit-time comparison method predict or describe noticeable quality defects, while others predict or describe service-level deficiencies such as latency that may not be noticeable to the user.

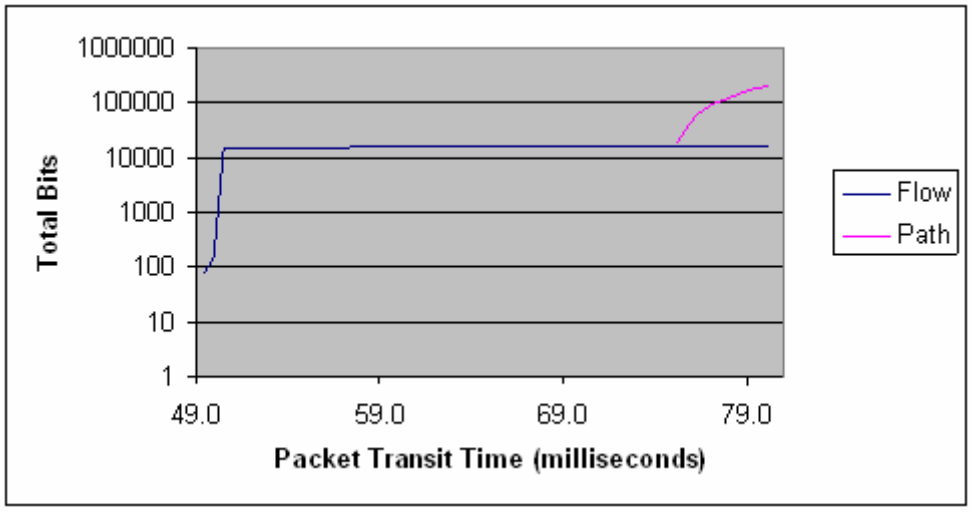

Figure 32. Comparison of Flow and Path Bit-Time Curves

In the second case, where five milliseconds of jitter were added to the constant 50-millisecond average latency, visible defects in video quality were apparent. The Iperf measurement is shown in Figure 33; due to technical issues with the computers used, this measurement was done subsequently with different computers. Although the effective throughput shown was actually higher than those in the baseline and first test cases, it can be considered analogous for the purposes herein. A marked increase in jitter is observed, though the throughput has increased and the loss has decreased due to the change in environment.

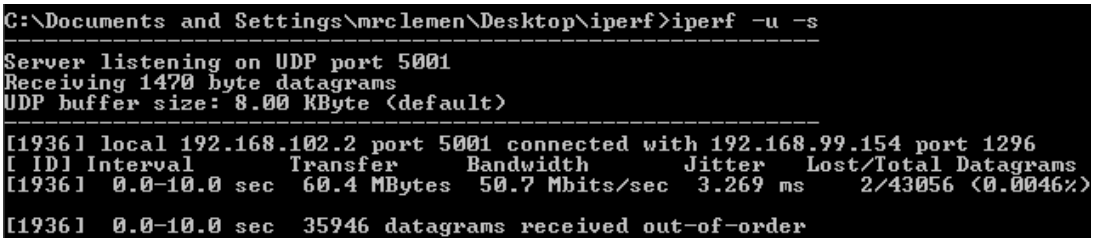

Figure 33. Results of Iperf Path Measurement with Jitter Added

Observing the video stream, there was noticeable frame loss as well as the introduction of artifacts in most frames, significantly obscuring the video. Interestingly, the path bit-time curve for this case does not predict these effects. Based on a ping test, the average measured latency was still 50 milliseconds, and the effective throughput should have been sufficient for the video. The only telltale sign that performance was degraded was the message in the Iperf results indicating that over 83 percent of packets were received out of order. For UDP video streams, packet arrival order is extremely important to video clarity; the introduction of stochastic jitter to a steady flow of data could easily cause this phenomenon and lead to quality defects. This was not reflected in the path bit-time curve or the bit-time comparison, which would look nearly identical to those in Figures 29 and 30, respectively. It is clear that there are additional performance factors the current formulation of link and path bit-time curves does not take into account. Possible revisions to the bit-time curve model will be discussed in the conclusions.

D. FIELD TESTING: ABILITY TO ARTICULATE SERVICES

1. Overview

The previous section tested the ability of the Bits in Time model, formulated as bit-time curves for flows and links, to describe and predict service performance for basic cases. This section complements that testing with a brief study of the ability of the CIR specification language described in the previous chapter to adequately specify actual information requirements or services that might be required in an operational environment. Due to the conceptual nature of the proposed model and the lack of existing

CIR monitoring tools, this section reports the results of a conceptual study done based on the network and application environment present in the Maritime Interdiction Operation (MIO) experiment conducted in September 2007. Although quantitative results were not attainable, the exemplar cases presented here portray actual services as understood from MIO experiment data.

2. MIO Experiment Network

MIO experiments are conducted in the San Francisco bay area, utilizing Yerba Buena Island (YBI) as a base of operations and local police and U.S. Coast Guard vessels for the maritime experiment platforms. The purpose of MIO experiments is to test ship-to-ship and ship-to-shore network technology along with collaborative information-sharing applications toward the end of increasing the speed of decision-making in maritime threat scenarios. These scenarios involve ships carrying suspect personnel and cargo approaching a port city, and the interdiction process that ensues. Various means of collaboration, including text, voice, and video, are combined with network-enabled radiological and biometric sensors to enable faster and more effective sharing of data between boarding personnel on the ships and subject matter experts on land.

From a physical network perspective, the MIO network consists of a backbone of broadband wireless links stretching across the bay area, ultimately tying into Internet connections at Lawrence Berkeley National Labs and Coast Guard Island. These sites also provide Virtual Private Network (VPN) tunnels back to NPS, and via NPS to several other partner organizations within the U.S. as well as in foreign countries including Sweden, Austria, and Singapore. During the September experiment, the primary operating areas were inside the bay and directly outside the Golden Gate Bridge, where teams on two interdiction vessels boarded respective target vessels to perform mock searches for radiological materials and suspected terrorists. An additional operating area on Mare Island, simulating riverine operations, was added for the first time. A high-level view of this infrastructure is provided in Figure 34.

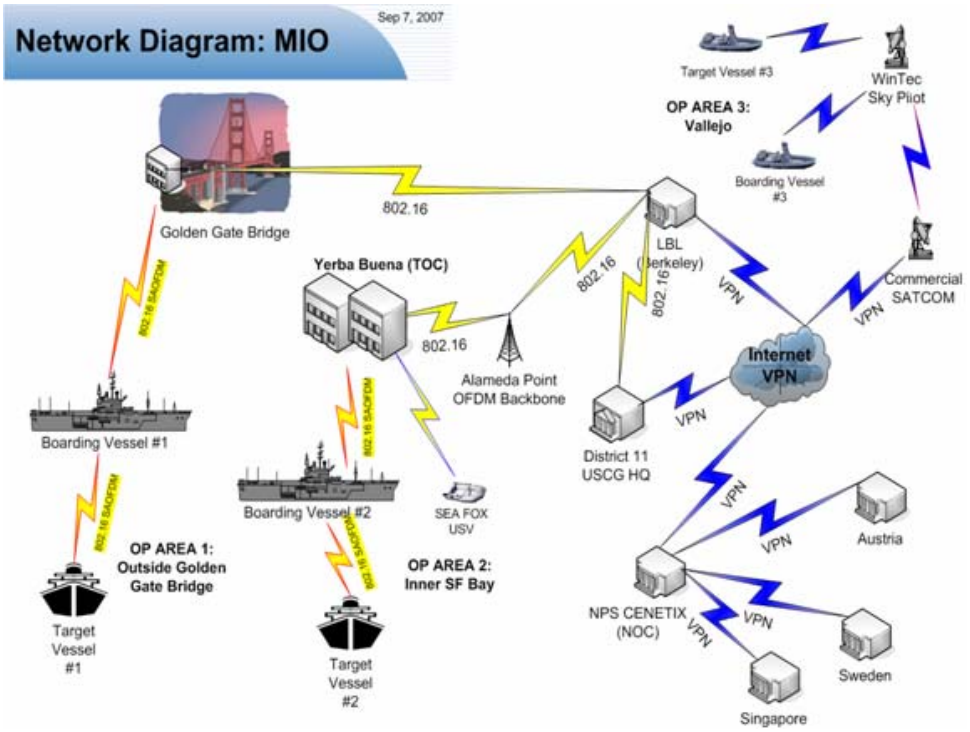

Figure 34. MIO Experiment Network Topology (From: Bordetsky et al. 2007)

From an applications perspective, the MIO focuses primarily on collaboration and information-sharing tools. Some of the desired information types include text chat, voice calls, live video, file sharing, and text-based discussion groups. Each of these functional capabilities is supported by one or more collaborative applications used within the MIO experiment. These tools connect the boarding parties that are performing the detection of nefarious cargo and persons with decision-makers and analysts on shore who can assist in processing the data collected throughout the interdiction experiment.

The specific information requirements dictated by each participant in these scenarios determines the set of CIRs that must be articulated to a holistic network management system in order to effectively monitor the services present and needed on during the interdiction. Two of these CIRs were chosen as exemplars for study. The first is Microsoft Groove, a collaboration tool that enables chat, file sharing, discussion groups, and many more information sharing mechanisms. Second is the set of live video feeds that provide real-time awareness to decision-makers away from the interdiction. Both of these are discussed in greater detail in the following sections.

3. Exemplar Case 1: Collaboration Suite

Groove (Groove Networks) is a collaboration suite designed to provide a virtual office environment for geographically-distributed teams. Each team may create one or more "workspaces," each of which constitutes an atomic collaborative space. A workspace may include a combination of collaboration and work management tools, such as task managers, discussion groups, file sharing, and group-wide text chat. Groove also provides a presence mechanism to inform a user of whom else is online or current using their workspace, and an individual-to-individual text messaging system. A screenshot of Groove in use during the September MIO experiment is given in Figure 35.

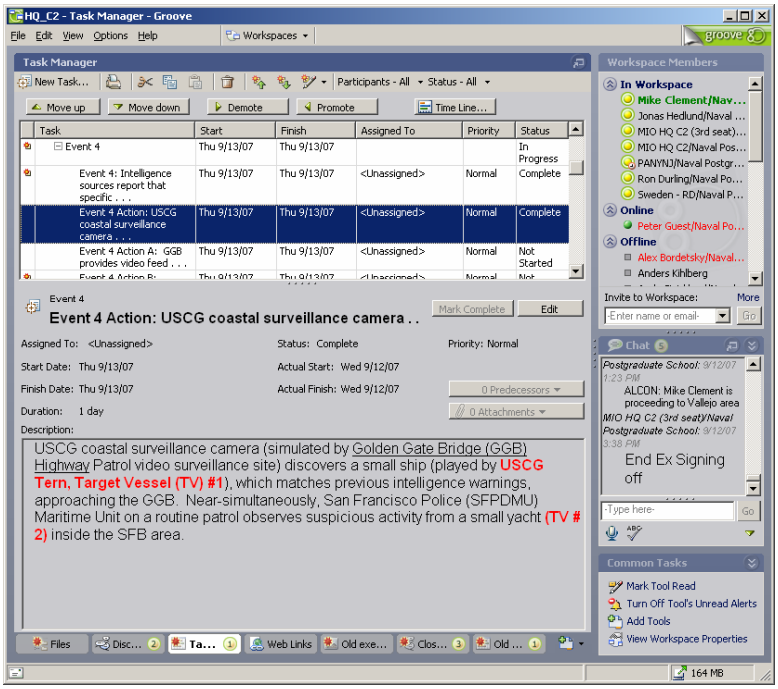

Figure 35. Groove in use during the MIO experiment

The underlying communications model utilizes both client-server and peer-to-peer channels to relay messages and updates to the virtual workspaces; each client synchronizes its current version of the workspace based on the latest updates from other members' workspaces. Although each user's current snapshot of the workspace is available offline, in order to maintain a synchronized state and to receive new text messages, both channels of connectivity must remain open. Therefore, Groove is an ideal

candidate for study in terms of a CIR: it consists of several underlying flows, each with its own parameters and requirements. An example of this topology is shown in Figure 36.

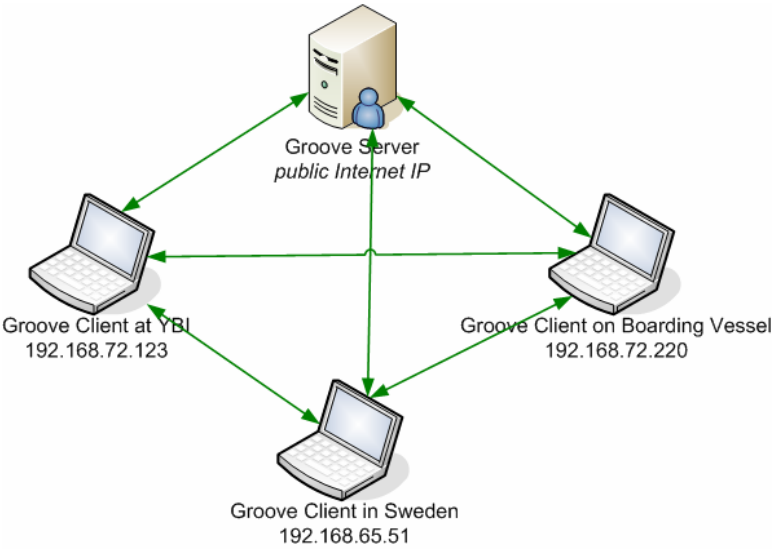

Figure 36. Groove Service Topology

Presume that the CIR for collaboration via Groove is to be defined for the user at YBI. The user must define a CIR that accurately describes both the service-level attributes and the flows required to achieve the desired collaboration. Relevant service-level attributes in this case would be reliability and responsiveness: reliability in terms of ensuring all updates to the workspace arrive, and responsiveness in terms of all updates arriving in a timely manner. Due to the nature of this service, reliability is a strict requirement; any bits that are lost are a detriment to the level of service. However, bits that are late may or may not negatively impact the level of service, depending on the user's need for those particular bits. For instance, an update to a shared file that the user does not need does not have a responsiveness requirement, but a single text message sent to that user containing urgent information may have very stringent responsiveness requirements. This may pose a challenge to describing the service with a single CIR.

Suppose that all workspace synchronization occurs via the client-server connection and has moderate responsiveness requirements. This means that all shared file updates, discussion group posts, and task manager status changes would be sent by the individual clients to the server, and then propagated to all other clients. In contrast,

suppose all individual-to-individual messages are sent peer-to-peer and have high responsiveness requirements. In this case, the flows may be distinguished by different bit-time curves; however, the challenge of unifying service-level attributes remains. Figure 37 depicts the user-required CIR inputs for this service.

```
<CIR
        user_id="YBI"
        cir_id="Groove"
        start="2007-09-10_17:00:00Z"
        stop="2007-08-12_21:30:00Z"
/>
        <Attribute
                type="Reliability"
                value="High"
        />
        <Attribute
                type="Responsiveness"
                value="High"
        />
        <Flows>
                <Flow
                        type="Burst.Data.Instant_Message"
                        remote="boarding1.vessels.uscg"
                />
                <Flow
                        type="Burst.Data.Instant_Message"
                        remote="remote_site.sweden"
                />
                <Flow
                        type="Transaction.Data.Record_Transfer"
                        remote="grooveserver.nps"
                />
        </Flows>
</CIR>
```

Figure 37. User-Required CIR Inputs for Groove Service

Assuming a translation mechanism as described in the previous chapter that can fill in the exact endpoints and service- and flow-level numerical variables, this should form a complete description of the Groove service. It depicts flows to each client for peer-to-peer traffic and a single flow to the server for workspace updates. The latter flow type is assumed to fall within the broader category of record transfers, much like an HTTP or DNS request: every time content in the workspace is updated, updates will be propagated to each client and acknowledgement of receipt will be returned. Since service-level attributes are scaling factors applied to each flow, this may be a sufficient description. However, one remaining issue is the complete enumeration of peer clients. A single Groove workspace can accommodate dozens of users, and users may be added to the workspace at any time. This being the case, a CIR that accounts for all Groove-related

traffic would itself have to be dynamic in order to describe all possible flows over time. It may be necessary to add a capability to this model to express a generic flow that does not have an explicit remote endpoint, but rather that expresses a bit-time curve that is the composite of a group of flows.

4. Exemplar Case 2: Live Video

Part of achieving situational awareness between geographically-separated sites within the experiment involved establishing live video links between each participant. Boarding parties with portable cameras fed video into online conference rooms so that decision-makers could see their live progress and gain a better understanding of the situation as it developed. Likewise, the command center posted its video along with several remote sites such as Sweden, enabling each participant to have improved awareness of the overall operation.

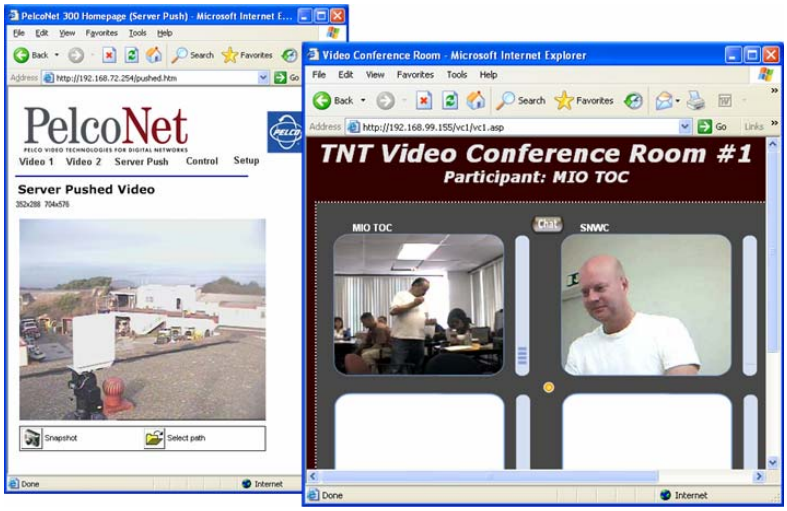

Figure 38. Video Sharing Tools

The two primary mechanisms used for video sharing were Pelco devices and internally-developed web-based virtual video conference room software; both are shown in Figure 38. Pelco devices were accessed directly via a webpage and used HTTP-encapsulated video streams. The conference room tool was centralized on a single server, which acted as a relay between senders and receivers. It too was accessed via HTTP, but

in this case the video was sent in Flash video format. This created a complex network structure ideal for study, a subset of which is shown in Figure 39.

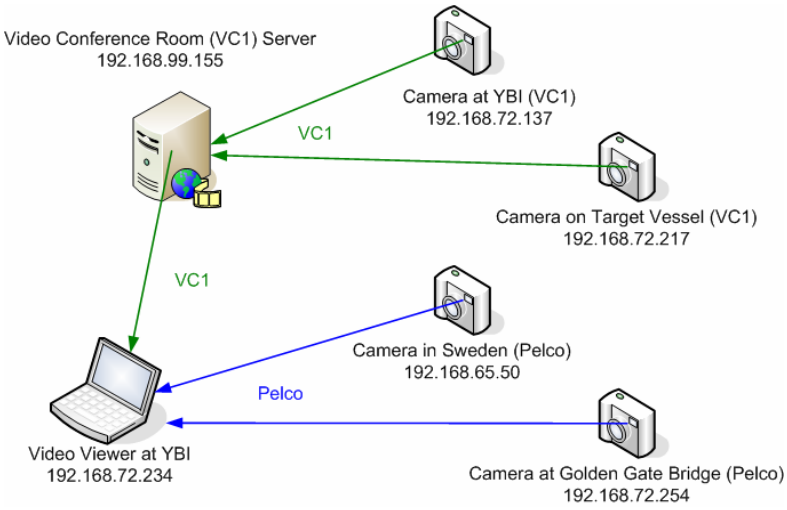

Figure 39. Video Service Topology

Although each video feed might comprise its own CIR in a real-world operation, this discussion assumes that the CIR is operational awareness in the form of live video of all participating sites. The service-level attributes of this CIR are uniform across all video streams to comply with the CIR specification model. As discussed in the example from the previous chapter, this CIR contains implicit flows that will not become stated until the user-inputted CIR is translated by the specification interface. For instance, both the Pelco and the video conference video feeds are accessed via web pages. It is not clear from the CIR shown in Figure 40 that these implicit flows exist. The flows from cameras to the video conference server are also implicit. Ideally, the flow type taxonomy would carry with it knowledge of these dependencies. Mapping flow dependencies and creating a taxonomy of flow types may be a suitable sub-topic for future research in this area.

```
<CIR
        user_id="YBI"
        cir_id="Video"
        start="2007-09-10_17:00:00Z"
        stop="2007-08-12_21:30:00Z"
/>
        <Attribute
                type="Clarity"
                value="High"
        />
        <Attribute
                type="Reliability"
                value="Moderate"
        />
        <Attribute
                type="Responsiveness"
                value="High"
        />
        <Flows>
                <Flow
                        type="Streaming.Video.Surveillance"
                        remote="pelco.remote_site.sweden"
                />
                <Flow
                        type="Streaming.Video.Surveillance"
                        remote="pelco.golden_gate"
                />
                <Flow
                        type="Streaming.Video.Surveillance"
                        remote="videoconf.nps"
                />
                <Flow
                        type="Streaming.Video.Surveillance"
                        remote="videoconf.nps"
                />
        </Flows>
</CIR>
```

Figure 40. User-Required CIR Inputs for Video Service

Since the example flow types used here are categorized by function and not by protocol, it is not clear that two of these video feeds are sent in higher-quality MPEG-4 video format and two are sent in lower-quality Flash video format. This makes a significant difference to the network management system as each protocol has its own resource requirements and thus different bit-time curves. It is unclear whether revising the taxonomy to specify the protocol is the better answer, or if an additional field in the translated CIR specification is needed to articulate this. Finally, like in the Groove example, it is possible for the number of clients in the video conference room tool to change over time. This reinforces the need to be able to articulate generic flow groups or classes that scale based on the number of active flows at any time.

E. FURTHER EXPERIMENTATION

The testing documented in this chapter raised several questions about the proposed model. Lab testing exposed uncertainties regarding the characterization of complex flow types and the assessment of flow performance in varying network conditions. Other challenges arose during the field testing, including combining widely varying flow types in a single CIR, sets of flows within a single service that change over time, creating a taxonomy of flow types that adequately describes differences between

similar protocols, and hidden dependencies between flows. These unanswered questions form starting points for future experimentation on the proposed model.

Future experiments might be categorized into two parts: Bits in Time model validation and refinement, and CIR specification validation and refinement. The model of characterizing, aggregating, and comparing bit-time curves requires further study to validate its accuracy in various conditions. Characterizing more complex flow types may expose areas of improvement to the model, simultaneously building the library of known flow types. Likewise, the CIR specification model should be tested in different environments and stretched to its limits, so that a refined model will emerge that has superior expressive capability. Taken together, these improvements should enable the proposed model to be a useful tool for holistic network management.

V. CONCLUSIONS AND FUTURE WORK

A. MAJOR CONCLUSIONS FROM RESEARCH

In an era of ever-increasing networking capability, both in terms of information available to the user and of resources available to carry that information, new ways of managing networks are becoming critical to day-to-day operations and to maintaining information superiority in the face of new threats. Traditional approaches that focus on delivering perfect service to a limited number of users over a fixed set of network resources cannot handle the imminent emergence of network-centric applications that operate in frequently imperfect conditions. Dealing with less than perfect service quality and managing a network where the set of available resources change on the order of minutes, not months, is the emerging business model for network operations in the tactical environment.

The aim of this thesis is to illuminate this issue and address the concept of holistic network management in a tangible, graspable way that can form a basis for further research in this burgeoning field. A model is presented that connects the highest level concept of a user-articulated service with the lowest level concept of bits with deadlines traversing the network. This model describes a concrete language for depicting a user's critical information requirements in terms that the user can understand, while still maintaining the descriptive elements critical to managing and monitoring those requirements. At the same time, it provides an algebra for expressing the atomic elements of the network configuration, flows and links, in terms of a single descriptor, and for both aggregating and comparing those descriptors in a way that allows assessment of the overall level of service.

It is clear that the work is far from finished. The specification of CIRs is rudimentary and requires improvements to address the issues raised in the previous chapter. Likewise, the bit-time curves proposed do a proficient job of articulating certain kinds of traffic requirements, but are not yet suited to many types of flows. The relationship between certain network effects and their corresponding curves also requires

clarification. However, the capacity of the model to express high-level requirements in a way that could be translated into lower levels and simultaneously to measure low-level metrics and generate a higher level assessment shows that this model has interesting properties that merit further study.

B. DIRECTIONS FOR FUTURE RESEARCH

This research work presents a framework for tying distinct flows of data together into user-oriented services, and for comparing network requirements of those services to the capability of the network infrastructure that carries those services. Many new concepts are introduced, including the mapping of service-level requirements onto network attributes, the Bits in Time model of network performance, and various measuring and visualization techniques. Each of these areas exposes additional topics open to future research work. Some of these topics are illustrated below for the consideration of those wishing to pursue further research in this area.

1. CIR Translation and Evaluation Frameworks

Although this research offers an interesting, if not novel, approach to describing and computing network performance, it leaves unresolved many of the difficult mathematical relationships involved in the algebra of bit-time curves. Other branches of mathematics and computer science may be incorporated into this model to address these challenges. One example is the application of fuzzy logic to network and service performance. (Zhang and Zhu 2005; Yaghmaei et al. 2006; Wang et al. 2006) all describe applications of fuzzy logic to QoS problems; it could also be formulated as "fuzzy" levels of service based on "fuzzified" network attributes. Essentially, fuzzy logic is a branch of mathematics that extends the notion of set theory by allowing a single entity to have partial membership of multiple sets. In other words, it is possible to have *somewhat* high bandwidth and simultaneously *mostly* medium bandwidth; the amount of latency partially belongs to both the high and low sets. Figure 41 illustrates the membership function for a fuzzy variable; this is adapted from an example given in (Crnkovic-Dodig).

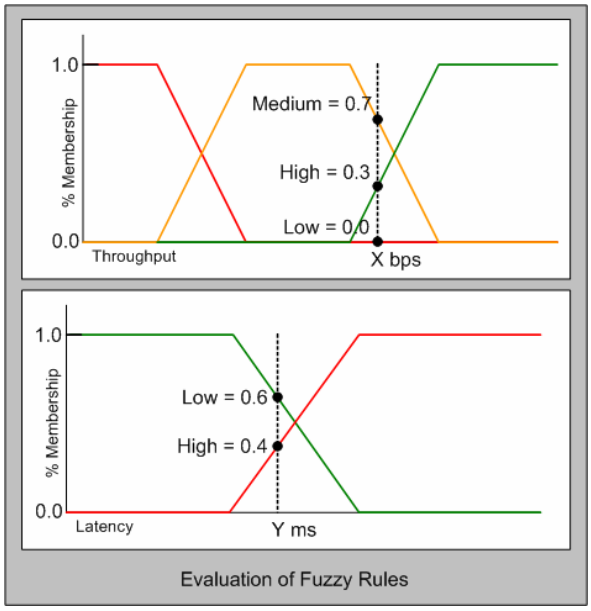

Figure 41. Examples of Fuzzy Membership Functions

There are two primary components and four main steps to a fuzzy logic problem. One component is the collection of membership functions for each input and output variable; the input variables are shown in Figure 41. The "crisp," or exact, values of throughput and latency are "fuzzified" into partial memberships of each fuzzy set during the first step of the process. Although the mapped fuzzy values are normalized in this example, normalization is not a requirement for fuzzy math. The other component is the set of rules for mapping input variables onto output variables, which are used during the evaluation step. Rules are standard "if-then" statements that map input sets onto output sets. The set of rules is shown in Table 5.

1. If Throughput is Low, then Performance is Poor
2. If Throughput is Medium and Latency is High, then Performance is Fair
3. If Throughput is High or Latency is Low, then Performance is Good

Table 5. Fuzzy Rules

Take the second rule; this uses two input variables to compute the output. In fuzzy set theory, "and" means to take the minimum of the two inputs, whereas "or" means to take the maximum. So in this case, since $\text{Thoughput}_{\text{Medium}} = 0.7$ and $\text{Latency}_{\text{High}} = 0.4$, $\text{Performance}_{\text{Fair}} = \text{Min}(0.7, 0.4) = 0.4$. Unlike rigid rules, where only the first or closest match is evaluated, all fuzzy rules are evaluated and then aggregated, as denoted by the highlighted region in Figure 42.

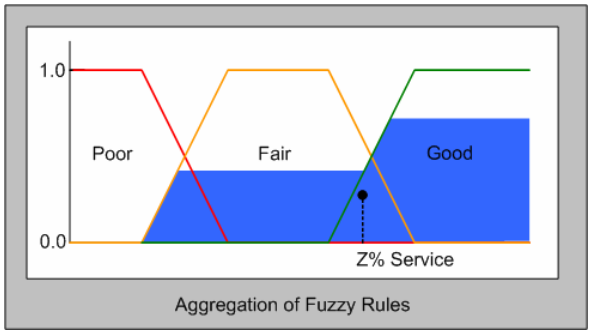

Figure 42. Aggregation of Evaluated Fuzzy Rules

Of course, finding this region does not by itself provide actionable information. Computers and humans alike operate best with something quantitative. In the final step, techniques such as taking the center of mass of the highlighted region are used to estimate the "crisp" value of each output variable.

An interesting piece of research would be to apply a fuzzy model of service to complement the Bits in Time model, and compare the projections against user perceptions of service. Like the Bits in Time model, fuzzy logic provides an attractive algebra for expressing multiple dimensions of network attributes. It is not clear how service requirements would be compared against network capability; defining such a comparative technique would itself be an interesting piece of work, possibly drawing on other related applications of fuzzy logic (Zhang and Zhu 2005; Yaghmaei et al. 2006; Wang et al. 2006).

Another relevant technique taken from computer science is the application of artificial neural networks and case-based reasoning to treat the complex relationships between bit-time curves as patterns with outcomes that can be learned. Bordetsky et al. (2003) discuss the introduction of feedback controls and case-based reasoning memory

into QoS-enabled multimedia networks. For instance, in networked teleconferences there is a Call Preparation Control mechanism that establishes the quality of connections during a teleconference session. In their paper, this mechanism is equipped with a case memory to learn from previous sessions how to recognize and respond to certain configuration patterns in order to achieve the best possible level of service. Applied to CIR translation and evaluation, these techniques may enable the holistic management system to recognize certain combinations of flow types, or certain network effects, and apply known patterns to the aggregation and comparison of bit-time curves. This might be tested in a way similar to that proposed above for fuzzy logic.

2. Tools for Selecting Service Quality Levels

Even with a hierarchical model that defines service-level attributes abstracted from the network level, asking a user to define their requirements in terms of clarity or reliability is difficult. Experience can help by providing templates for certain services in certain operational contexts, but there is still room for a friendly user interface for specifying these requirements. Imagine an intelligence imagery feed; live pictures of a monitored site with known high-value targets are fed across the network to a remote command center. Although having high-resolution fast-updating imagery would be anyone's desire, the acceptable minimum level of service depends on the usage of that imagery. For instance, will it be necessary to identify any of the entities in the image, such as people or vehicles? Is it more important to know the fact that a person entered or left the building, or is important to see which direction they walked to get in or out of it?

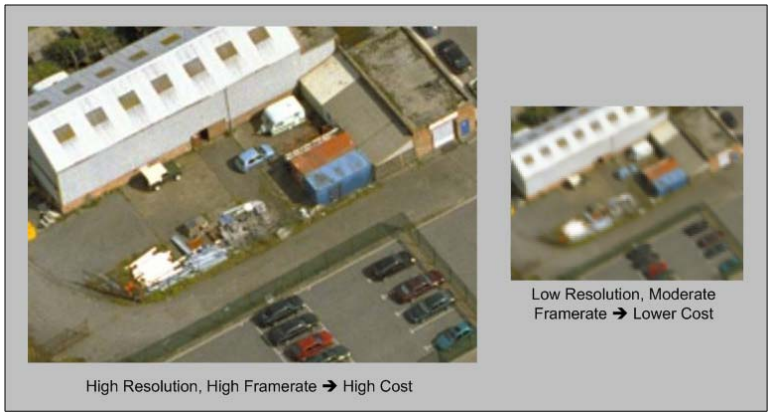

Figure 43. Criteria for Varying Quality of a Video Stream

As shown in Figure 43, different levels of service allow the user to see different things, and there is a cost associated with each level according to the network resources it consumes. It may be useful to build a user interface that presents multiple versions of a particular service, depicting both the difference in quality and the cost of each version. If the cost is portrayed in terms that matter to the user (e.g., ability to support their services as well as conduct conference calls, et cetera), such a tool may be useful in convincing users to select reasonable vice best case settings. This tool could be defined for different types of service, calibrated to users' needs in a variety of contexts, such as surveillance, command and control, collaborative communications, and so forth.

3. Improving Measurement Techniques

There is a lot left to be done in this area. This research proposes the infrastructure for measurement, but leaves many aspects open-ended. One area of contribution is network topology discovery. All experimental work done herein required a priori knowledge of every link in the network. In future network-centric operations, the task of mapping every link within a single, highly-mobile unit, let alone the complex interconnections between joint and coalition forces, would take a heroic effort. Offloading this task onto autonomous distributed agents that scan the network, noting both topology and performance characteristics, would greatly accelerate the process of calibrating the service performance model to the network at hand. There are many existing protocols and techniques for general-purpose and task-specific network

discovery. Cisco Systems uses their own discovery protocol for finding the connections between switches and routers; many routing protocols can automatically converge a large set of interconnected routers into a coherent routing tree. Evaluating and synthesizing techniques for this application is a necessary task for making holistic network management feasible in real-world networks.

Accurate measurement of link and path performance attributes is another area worthy of further study. As pointed out in (Prasad et al. 2003), there are a variety of techniques for measuring network capability, each with its own level of accuracy and cost in network transmissions; however, some measurements, such as per-link available bandwidth, are still elusive. Many network characteristics can be measured for an entire path, or per link, just using the path endpoints. Certain per-link measurements can only be assessed by agents installed at intermediate routers. Determining the minimal set of measurements necessary to determine the capability of the network, and defining the appropriate measurement infrastructure, is a useful research task.

The proposed model was created with certain kinds of services in mind, such as surveillance video and text messaging. These services utilize flows that are either constant, or statistically periodic. Other services consist of complex sets of flows that are difficult to express with the Bits in Time model and other similar methods. The Groove office collaboration tool, which is used heavily in TNT and MIO, is a good example. Groove is centered on the synchronization of files, discussion boards, whiteboards, et cetera between individual instances of the application. The network requirements are entirely dependent on the user activity within each instance of the Groove workspace, making a statistical description very difficult. Services where the service-level quality attributes change frequently or depend heavily on the specific sub-task at hand are also difficult to depict with such models. Testing the Bits in Time model with these types of services and extending the model as needed adds versatility to the technique.

4. Visualizing Service Performance

The best network management model in the world is not very useful to users if all they have to look at are lists of number without any meaningful context. Creating a

graphical interface that portrays service performance in a way that is both meaningful and actionable is arguably just as valuable as the management model underlying it. Current representations of network state and performance focus on network reach and link utilization. Reach is often represented with red-light, green-light indicators, either in a listing of important nodes or overlaid on an image depicting the network topology. Solarwinds (Solarwinds) takes this approach. Link utilization is generally portrayed via speedometer-like graphics, or maybe color-coding applied to links in a graph that illustrates the network.

However, when the important data are performance metrics for services between many machines across a complex network, with each endpoint having many connections to other endpoints, and each type of flow running between several pairs of endpoints simultaneously, portrayals using meters and color-coding are hardly sufficient. The prototype interfaces used for testing in this thesis are only meant to demonstrate the kinds of data that would be used by the proposed management model. There is a whole thesis worth of research to be done strictly on how to graphically represent a network of services, each of which contains one or more flows with their respective network performance characteristics.

5. Service Adaptation

Raising the level of network management from a bits-focused view to a services-focused view is a major step toward holistic network management. Once the ability to monitor services exists, the next logical step is to apply this newfound service awareness to intelligent management of network resources. Adaptive networking is a field of study that focuses on adapting both the network configuration and individual flows in order to achieve a state of acceptable performance for every user on the network.

An overview of adaptive networking is presented in (Clement and Bordetsky 2006). Adaptive networking approaches vary, but generally fall into categories. First, there are techniques that focus more on adapting flows to match available network resources by gracefully degrading their quality. For instance, the resolution or frame-rate of a video feed may be reduced so that the network requirements of the flow are within

the available capacity of the network path. Other approaches focus on adapting the behavior of the network to better accommodate the requirements of the services present. This may involve combining similar flows along the same path into one flow that meets each flow's demands, such as two requests for the same video, but each with different resolution requirements.

Adaptive networking techniques may also be categorized by the level of involvement on the part of the applications and the users. Some approaches are application-aware; that is, the application itself negotiates with the network for resources and adapts its behavior according to the resources allocated to it. Application-transparent approaches provide the same functionality, except that network agents negotiate on behalf of the applications, and adapt application traffic in a way that is "transparent" to that application. Human-aware techniques place users in charge of making decisions about their own usage, informing them of the constraints of the network to provide them with services.

A study of these approaches, and the benefits of combining these with holistic network monitoring techniques, would take network management to a new level, enabling users to specify their needs and allowing the network to intelligently manage and provide the best possible service even when all requirements cannot be fully satisfied.

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

Alberts, David S., John J. Garstka, and Frederick P. Stein. 1999. *Network Centric Warfare: Developing and Leveraging Information Superiority*. 2nd ed. Washington, DC: CCRP Publications.

Barford, Paul, et al. 2001. On the Marginal Utility of Network Topology Measurements. Paper presented at the 1st ACM Special Interest Group on Data Communications, November 1-2, in San Francisco, USA.

Bauer, Ben and Andrew S. Patrick. 2004. A Human Factors Extension to the Seven-Layer OSI Reference Model. http://www.andrewpatrick.ca/OSI/10layer.html (accessed March 13, 2006).

Bordetsky, Alex, Kevin Brown, and Leann Christianson. 2003. Adaptive Management of QoS Requirements for Wireless Multimedia Communications. *Information Technology and Management* 4 (January): 9-31.

Bordetsky, Alex et al. 2007. TNT MIO 07-4 Planning Document.

Caceres, Ramon, et al. 2000. Measurement and Analysis of IP Network Usage and Behavior. *IEEE Communications Magazine* 38 (May): 144-151.

Case, Jeffrey D., et al., 1990. A Simple Network Management Protocol (SNMP). IETF RFC 1157, May.

Choi, Yong-Hoon and Iksoon Hwang. 2005. In-service QoS monitoring of real-time applications using SM MIB. *International Journal of Network Management* 15: 31-42.

Claise, Benoit, et al. 2004. Cisco Systems NetFlow Services Export Version 9. IETF RFC 3954, October.

Clark, Allan and Stephen Gilmore. 2006. Evaluating Quality of Service for Service Level Agreements. Paper presented at the 11th International Workshop on Formal Methods for Industrial Critical Systems, August, in Bonn, Germany.

Clement, Michael and Alex Bordetsky. 2006. Systematic Adaptation to Network Resource Constraints in Coalition C2 Environments. Paper presented at the 11th International Command and Control Research and Technology Symposium, September 26-28, in Cambridge, England.

Combs, Gerald. Wireshark: The World's Most Popular Network Analyzer. Wireshark. http://www.wireshark.org/ (accessed September 5, 2007).

Crnkovic-Dodig, Luka. Fuzzy Math, Path I, The Theory. Peltarion. http://blog.peltarion.com/2006/10/25/fuzzy-math-part-1-the-theory/ (accessed August 15, 2007).

DaSilva, Luis A. 2000. QoS Mapping along the Protocol Stack: Discussion and Preliminary Results. Paper presented at the 2000 IEEE International Conference on Communications, June 18-22, in New Orleans, USA.

Deri, Luca and Stefano Suin. 2000. Effective Traffic Measurement Using ntop. *IEEE Communications Magazine* 38 (May): 138-143.

Estan, Cristian and George Varghese. 2001. New Directions in Traffic Measurement and Accounting. Paper presented at the 2001 ACM SIGCOMM Internet Measurement Workshop, November 1-2, in San Francisco, USA.

Exposito, Ernest, et al. 2002. XQoS: A Quality of Service Specification Language. Paper presented at the 2002 IADIS International Conference on WWW/Internet, November 13-15, in Lisbon, Portugal.

Galetska, Michael. 2004. User-Perceived Quality of Service in Hybrid Broadcast and Telecommunication Networks. Paper presented at the 5th Workshop on Digital Broadcasting, September 23-24, in Erlangen, Germany.

Ghetta, Riccardo and Juan Toledo. EtherApe, a Graphical Network Monitor. SourceForge.net. http://etherape.sourceforge.net/ (accessed September 4, 2007).

Groove Networks. Groove Virtual Office. Groove. http://www.groove.net/home/index.cfm (accessed June 12, 2007).

Guo, Xinping and Colin Pattinson. 1997. Quality of Service Requirements for Multimedia Communications. Paper presented at Time and the Web, June 19, in Staffordshire, England.

Huang, Dongjie and James Shi. 2001. Quality of Service Scheduling in Wireless Multimedia Communications. Paper presented at the 53rd Vehicular Technology Conference, May 6-9, in Rhodes, Greece.

International Engineering Consortium. 2007. Element Management Systems (EMSs): The TMN FCAPS Model of OSS Tasks. International Engineering Consortium. http://www.iec.org/online/tutorials/ems/topic03.html (accessed September 16, 2007)

International Telecommunications Union. 2001. End-user multimedia QoS categories. ITU-T G.1010, November.

Jacobson, Van. 1997. Pathchar: A Tool to Infer Characteristics of Internet Paths. Paper presented at the 1997 Mathematical Sciences Research Institute Conference, April 21, in Berkeley, USA.

Keen, Peter G. W. and J. Michael Cummins. 1994. *Networks in Action: Business Choices and Telecommunications Decisions*. Belmont: Wadsworth.

Keshav, S. and R. Sharma. 1998. Achieving Quality of Service through Network Performance Management. Paper presented at the 8th International Workshop on Network and Operating Systems Support for Digital Audio and Video, July 8-10, in Cambridge, England.

Kessler, Gary C. and Steven D. Shepard. 1997. A Primer On Internet and TCP/IP Tools and Utilities. IETF RFC 2151, June.

Loguinov, Dmitri and Hayder Radha. 2001. Measurement Study of Low-Bitrate Internet Video Streaming. Paper presented at ACM SIGCOMM Internet Measurement Workshop 2001, November 1-2, in San Francisco, USA.

Ma, Alex. Otter: Tool for Topology Display. Cooperative Association for Internet Data Analysis. http://www.caida.org/tools/visualization/otter/ (accessed September 4, 2007).

McCloghrie, Keith and Marshall T. Rose. 1991. Management Information Base for Network Management of TCP/IP-based internets: MIB-II. IETF RFC 1213, March.

Nahrstedt, Klara and Jonathan Smith. 1994. A Service Kernel for Multimedia Endstations. Paper presented at the 2nd International Workshop on Multimedia: Advanced Teleservices and High-Speed Communication Architectures, September 26-28, in Heidelberg, Germany.

Network Uptime. Free Enterprise Network Monitoring Tools – Big Brother. Network Uptime. http://www.networkuptime.com/tools/enterprise/big_brother.html (accessed September 4, 2007).

NIST Net. NIST Net Home Page. National Institute of Standards and Technology. http://www-x.antd.nist.gov/nistnet/ (accessed September 5, 2007).

O' Neil, Timothy M. 2002. Quality of Experience and Quality of Service: For IP Video Conferencing. White Paper by Polycom.

Parulkar, Guru, et al. 1997. An Architecture for Monitoring, Visualization, and Control of Gigabit Networks. *IEEE Network*, vol. 11 (September/October): 34-43.

Pelco Corporation. Pelco website. Pelco Corporation. http://www.pelco.com/ (accessed September 5, 2007).

Prasad, Ravi, Constantinos Dovrolis, Margaret Murray, and KC Claffy. 2003. Bandwidth Estimation: Metrics, Measurement Techniques, and Tools. *IEEE Network*, vol. 17 (November/December): 27-35.

Senge, Peter. 2006. *The Fifth Discipline: The Art and Practice of the Learning Organization*. Rev. ed. New York: Currency.

Siller, Mario and John Woods. 2003. Improving Quality of Experience for Multimedia Services by QoS arbitration on a QoE Framework. Paper presented at the 13th Packed Video Workshop, April, in Nantes, France.

Solarwinds. Network Management Software. Solarwinds. http://solarwinds.com/ (accessed July 26, 2007).

Strauss, Jacob, Dina Katabi, and Frans Kaashoek. 2003. A Measurement Study of Available Bandwidth Estimation Tools. Paper presented at Internet Measurement Conference 2003, October 27-29, in Miami Beach, USA.

Tirumala, Ajay, et al. NLANR/DAST: Iperf. National Laboratory for Applied Network Research. http://dast.nlanr.net/Projects/Iperf/ (accessed September 6, 2007).

Van Creveld, Martin. 1985. *Command in War*. Cambridge: Harvard University Press.

VLC Team. VLC Media Player. VideoLAN. http://www.videolan.org/vlc/ (accessed September 5, 2007).

Wang, Ping, et al. 2006. A Fuzzy Model for Selection of QoS-Aware Web Services. Paper presented at the 2006 IEEE International Conference on e-Business Eingineering, October 24-26, in Shanghai, China.

Yaghmaei, M., M. Baradaran, and H. Talebian. 2006. A Fuzzy QoS Routing Algorithm for Communication Networks. Paper presented at the 10th IEEE International Conference on Communication Systems, October 30 – November 1, in Singapore.

Zhang, Runtong and Xiaomin Zhu. 2005. Fuzzy Routing in QoS Networks. In *Lecture Notes in Computer Science*, ed. L. Wang and Y. Jin, Vol. 3614: 880-90. Berlin: Springer-Verlag.

Zhou, Chen, Liang-Tien Chia, and Bu-Sung Lee. 2005. Semantics in Service Discovery and QoS Measurement. *IT Professional* 07, no. 2 (March/April): 29-34.

Zimmerman, Hubert. 1980. OSI Reference Model: "The OSI Model of Architecture for Open Systems Interconnection." *IEEE Transactions on Communications* 28: 425-32.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, VA

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, CA

3. Dr. Alex Bordetsky
   Naval Postgraduate School
   Monterey, CA

4. Lt. Col. Karl Pfeiffer, USAF
   Naval Postgraduate School
   Monterey, CA

5. Dr. Dan Boger
   Naval Postgraduate School
   Monterey, CA

6. Dr. David Netzer
   Naval Postgraduate School
   Monterey, CA